Structural Plasticity in Neuromorphic Systems

Dissertation zur Erlangung der naturwissenschaftlichen Doktorwürde

(Dr. sc. nat.)

vorgelegt der

Mathematisch-naturwissenschaftlichen Fakultät

der

Universität Zürich

von

Richard Miru George

aus

Deutschland

Promotionskommission

Prof. Giacomo Indiveri, PhD (Vorsitz)
Prof. Valerio Mante, PhD
Prof. Stefano Vassanelli, MD PhD

Zürich, 2018

Disclaimer

I hereby declare, that the work in this thesis is that of the candidate alone, except where indicated in the text, and as described below. Chapter 1 contains a review of relevant biological foundations that the later chapters are based on. The work described here was conducted by others, as indicated by the corresponding references. The conclusions drawn on the presented findings are made by the candidate, except stated otherwise. Every part of the project was shaped by discussions with Giacomo Indiveri, Stefano Vassanelli, Christian Mayr and Peter Diehl. Marta Maschietto provided experimental data on functional connectivity in neuron cultures in vitro. This data is presented in Appendix A.7. As a Bachelors project, Robin Spiess helped with the simulations of structural plasticity on conventional hardware. Results of this collaboration are published in Spiess et al. [2016]. The work by Deger et al. [2016], which relies on mechanisms that are similar to those presented here, took inspiration from personal correspondence with one of the authors of [George et al., 2015b]. The appendix contains a paraphrased except from a paper on the creation of a distributed biohybrid neuromorphic setup, (See Appendix A.6) written in collaboration with A.Serb, C.Mayr and T.Podromakis., that is currently under submission in Nature Nanotechnology. Also the section 'Evaluation of Stochasticity in Neuromorphic Synapses' (Appendix A.3) is a paraphrased version of George and Indiveri [2016].

The use of 'we' in this thesis refers to me, the candidate, but also to the input received from the above-mentioned people.

Acknowledgements

I have to thank many people for their support and contributions to this work.
First and foremost, my sincere thanks go to my supervisor Giacomo Indiveri who gave me the opportunity to work on this project and supported me continuously with his ideas and expertise.
Furthermore I would like to thank Stefano Vassanelli, for his role in the *RAMP* project and his support. I am grateful to Christian Mayr, Peter Diehl and Taras Iakymchuk, who significantly influenced my work through their feedback and collaboration.
I am also very thankful to all the participants in the *RAMP* project, in particular to Alex Serb, Erica Covi, Marco Renato and Andrea Corna - Working with you has been a pleasure.
Thanks to Adrian Whatley and Manu Nair the readability of the presented document was improved significantly.

My thanks go also to all the people that make the Institute of Neuroinformatics what it is, and the mensa staff that fuelled this work with their coffee.
This research has received funding from the European Union 7th Framework Programme (FP7/2007- 2013) under grant agreement no. 612058 (RAMP).

Abstract

Modern state-of-the-art biohybrid systems provide closed-loop interactions between biological and artificial subsystems. In this context, the processing of electrophysiological data has to be performed in biological real-time to lead to adequate feedback stimulation. Conventional computing paradigms require massive resources for processing the acquired data due to their sequential and synchronous nature when faced with information coming from a massively parallel and asynchronously operating population of cells. Potential candidates for a more efficient interaction are neuromorphic processors. Neuromorphic analog VLSI emulates biophysical processes of neural tissue using CMOS transistors operated in the sub-threshold regime and features biologically plausible levels of parallelism. This allows the implementation of spiking neural networks for the processing of biosignals in hardware. One of the constraints that however limits the scalability of neuromorphic information processing architectures is the number of available synapse circuits. This number is naturally limited by the integrated circuits' two-dimensional layout. This constrains the ability to interact with large numbers of information sources like electrodes in large-scale arrays. To alleviate this limitation, we investigate strategies used by biological systems to solve related problems: Networks of biological neurons undergo a constant reconfiguration of their topology via activity-dependent plasticity mechanisms. The observed dendritic spine dynamics can be hypothesized to implement a strategy to limit the number of synapses that are not contributing to the performance of

the network. The possibility of using structural plasticity as a biologically-inspired strategy for optimizing resource usage in neuromorphic processors is explored in this project. In this thesis, I present an algorithm that allocates a limited number of synapses during runtime to choose event-sources that best contribute to the postsynaptic neuron's activity. This algorithm is implemented on a mixed analog/digital VLSI system and evaluated. In this context, neuronal activity can serve as an indicator of which synapse to connect to which source, mimicking activity dependent dynamics of dendritic spines and making optimal use of the available resources on the neuromorphic hardware. The evaluation promises increased scalability and a novel approach to expanding the number of input channels to neuromorphic systems. The results of this work suggest that the approach taken opens new perspectives for memory optimization in the simulation of spiking neural networks in general and in neuromorphic hardware in particular.

Zusammenfassung

Der aktuelle Stand der Technik in der Entwicklung biohybrider Systeme erlaubt die Erzeugung eines geschlossenen Informationskreislaufs zwischen einem biologischen- und einem technischen Sub-System. Konventionelle Paradigmen der Informationsverarbeitung
schränken in ihrer sequenziellen und synchronen Natur das potenzial solcher Systeme ein, wenn es um die Verarbeitung von massiv parallelen und asynchronen Datenströmen aus biologischen Zellkulturen oder lebenden Organismen geht. Potenzielle Kandidaten für eine nahtlose Interaktion, sind neuromorphe Prozessoren welche mittels CMOS Transistoren im subthreshold Regime, neuronale Prozesse emulieren um eine hohe Energieeffizienz und ein biologisch plausibles Level an Parallelität zu erreichen. Eine der Einschränkungen, welche jedoch die Skalierbarkeit von neuromorphen Prozessoren limitiert, ist die Anzahl von verfügbaren synaptischen Schaltungen. Diese ist vorgegeben durch die Dimensionen der Gesamtschaltung. Auch die Fähigkeit der Interaktion mit einer grossen Anzahl an Informationsquellen wie in etwa Elektroden in grossformatigen Arrays ist so beeinträchtigt. Um die Resourcennutzung in neuromorphen Prozessoren zu optimieren, betrachten wir, wie biologische Systeme ähnliche Probleme lösen. Neueste neurowissenschaftliche Erkenntnisse legen nahe, dass biologische Neuronen einer konstanten Umstrukturierung ihrer Topologie durch aktivitätsabhängige Plastizitätsmechanismen unterliegen. Das beobachtete Ausbilden und Zurückziehen von dendritischen Dornfortsätzen, stellt möglicherweise eine Strategie

dar, welche die Anzahl von Synapsen minimiert, die nicht zur Leistungsfähigkeit des Netzwerkes beitragen. Wir behandeln die Möglichkeit, strukturelle Plastizität als eine Strategie der Resourcenoptimierung zu nutzen. Der vorgestellte Mechanismus setzt während Laufzeit eine limitierte Anzahl an Synapsen ein um Eventquellen auszuwählen, die bestmöglich zur postsynaptischen neuronalen Aktivität beitragen. Durch das Nachahmen aktivitätsabhängiger Dynamiken dendritischer Dornfortsätze, kann neuronale Aktivität als ein Indikator dafür dienen, welche Quelle mit welcher Synapse zu verbinden ist, wodurch die Nutzung vorhander Resourcen neuronaler Hardware optimiert wird. Das Modell struktureller Plastizität wurde speziell für den Einsatz auf einem, im Rahmen dieser Arbeit entwickelten, digitalen Coprozessor entwickelt. Die beschriebene Evaluierung verspricht eine erhöhte Skalierbarkeit, sowie einen neuartigen Ansatz für eine erhöhung der Anzahl von Informationskanälen in neuromorphen Systemen. Darüber hinaus zeigen wir das der verwendete Ansatz Perspektiven der Speicherpotimierung in der Simulation von 'Spiking Neural Networks' im allgemeinen eröffnet.

Table of Contents

List of Figures

Acronyms

AER Address-Event Representation

ARM Advanced RISC Machine

CMOS Complementary Metal-Oxide-Semiconductor

DVS Dynamic Vision Sensor

EPSC Excitatory Post-Synaptic Current

FPGA Field Programmable Gate Array

I/O Input/Output

IPSC Inhibitory Post-Synaptic Current

IP Intellectual Property

LFP Local Field Potential

LTD Long Term Depression

LTP Long Term Potentiation

NMDA N-Methyl-D-Aspartate

PCB Printed Circuit Board

RAM Random Access Memory

ROLLS Reconfigurable On-Line Learning Spiking

SRAM Static Random Access Memory

STDP Spike-Timing Dependent Plasticity

STP Short-Term Plasticity

USB Universal Serial Bus

VHDL VHSIC Hardware Description Language

VLSI Very Large Scale Integration

AMPA α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid

AMPAr α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid receptor

AXI Advanced eXtensible Interface

CaMKII calmodulin-dependent protein kinase II

DIV Day in vitro

EDK Embedded Development Kit

EEG Electroencephalography

EOSC Electrolyte-oxide-semiconductor capacitors

EOSFET Electrolyte-oxide-semiconductor field-effect transistor

FiFo First-In-First-Out

fMRI Functional Magnetic Resonance Imaging

GCC GNU Compiler Collection

IC Integrated Circuit

ID identifier

ISR Interrupt Service Routine

RISC Reduced Instruction Set Computer

SoC System-on-Chip

pSoC programmable System-on-Chip

JTAG Joint Test Action Group

LFSR Linear Feedback Shift Register

MEA Multielectrode Array

MDM Microprocessor Debug Module

MHS Microprocessor Hardware Specification

MSS Microprocessor Software Specification

NMDA N-methyl-D-aspartate

NMDAr N-methyl-D-aspartate receptor

PAER Parallel Address Event Representation

PC Personal Computer

PSD Postsynaptic Density

PSD-95 postsynaptic density protein 95

RAMP Real neurons-nanoelectronics Architecture with Memristive Plasticity

SAP-97 synapse-associated protein 97

SR Shift Register

UDP User Datagram Protocol

VGCC L-type voltage-gated calcium channel

Wnt Wingless-related integration site

XST Xilinx Synthesis Technology

XSDB Xilinx System Debugger

1

A Neuromorphic Engineering Approach to Biohybrid Systems

The question of how brains perceive the environment and use the extracted information to form decisions which guide behavior has been the source of thousands of years of debate among philosophers and physicians. Already 300 BC, Herophilos of Chalcedon distinguished sensory and motor nerves [Pearce, 2013]. In light of this, computational neuroscience is a relatively young field that aims to answer these questions through the means of scientific methodology. Already today, computational principles identified through this approach, have made their way into new technologies such as deep neural networks and computer vision systems.

An important step in advancing knowledge in the domain of neuroscience is the development of techniques that allow the investigation of neuronal populations at different scales from the single neuron to whole brains.

Biohybrid systems like brain-machine interfaces go one step further in providing a tool for interacting with, rather than for the passive observation of, populations of neurons.

In this way, systems are created, that consists of both artificial and biological parts. Establishing a connection between

these parts opens up novel perspectives in neuroprosthetics, where one day neuronal tissue, damaged by neurodegenerative diseases, or strokes, could be substituted with artificial systems that are functionally equivalent.

The interaction with populations of neurons has been realized on very different spatial and temporal scales, through various signal acquisition and stimulation techniques:

- Functional Magnetic Resonance Imaging (fMRI) studies acquire the blood-oxygenation response on a whole brain scale in response to an external stimulus. [Sulzer et al., 2013]

- Electroencephalography (EEG) trades off spatial resolution with an increased temporal resolution against investigation of evoked potentials.

Both techniques are employed in neurofeedback experiments where the acquired signal shapes the sensory stimulus to the subject.

- In in-vitro, various patch clamp techniques allow intracellular electrophysiological data acquisition on a single neuron level. Dynamic clamp [Sharp et al., 1993] provides feedback that drives the neuron's membrane potential.

Advances in integrated circuit technology allowed the emergence of MEA as another in vitro method that we are going to focus on as an input to a neuromorphic system.

1.1 Multi-Electrode Arrays

Silicon-based MEA allow the recording and stimulation of hundreds of cells that are directly cultivated on top of the sensor to provide an extracellular interface to the neuronal population. In order to reduce the number of glia-cells, typically, the neuronal tissue is centrifuged, and the remaining dissociated embryonic cells are cultivated directly on the surface of the sensor. The sensor surface is pharmacologically processed to allow cells to grow in a laminar fashion in a Perspex chamber.

Somata, which provide the largest amplitudes when the neuron fires, sparsely cover the sensor's surface. Over the course of hours to days, growth processes change the coverage of the sensors, and the formation of excitatory and inhibitory synapses among the cultured neurons influences the firing statistics recorded at the electrode sites.

The high spatial resolution of advanced EOSC and EOSFET Multi-Electrode Arrays of 2048 electrodes in an active area of $4.48 \times 2.43 mm^2$, at a sampling rate of up to $20 kHz$, allows the electrophysiological investigation of the functional network within a neuron population, on a single-cell level [Dragas et al., 2017]. In the referenced work, an additional neurotransmitter and Local Field Potential (LFP) monitoring is made available. Moreover, the combination with additional optical studies of the culture's morphology through staining techniques is possible, to give insight into synapse formation over time through statistical analysis of the expression of markers such as synapsin.

The capacitive electrode-electrolyte interface (Figure 1.1) is typically provided through electrodes that directly connect to transistors gates to form EOSFETs [Vassanelli, 2014]. The sensor surface that is formed in this way, is covered by a dielectric layer post-processed after the standard Complementary

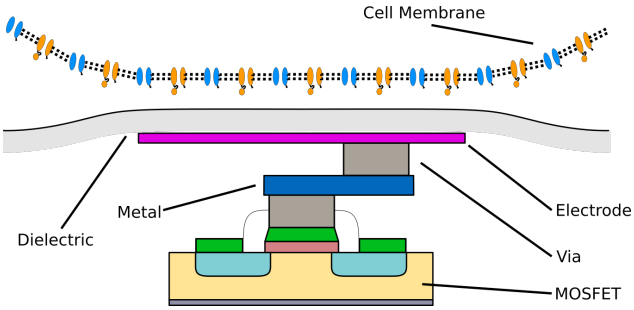


Figure 1.1:
Illustration of an EOSFETs operating principle. The electrode is directly connected to the gate of the transistor, sensing changes in the extracellular potential and causing a proportional current through the transistors channel.

Metal-Oxide-Semiconductor (CMOS) process. As the extracellular ion concentration changes, the potential on the gate changes. The resulting currents between the source- and drain terminal of the EOSFET can be processed by subsequent circuits on the same chip. Pre-processing typically involves preamplification and filtering, after which the signals are multiplexed before passing a second processing stage and subsequent conversion from analog to digital values. The fact that already at the lowest level of signal acquisition, multiplexing is used, gives rise to the possibility to optimize data throughput by hand-picking which channels to record from.

In the case of EOSC stimulators (Figure 1.1), the application of stimulation waveforms on the electrode forms an electric field which extends through the tissue and causes the local manipulation of ion concentrations to affect the membrane potential of the neurons present in the substrate. The TiO_2/ZrO_2 passivation, which acts as a dielectric, limits the

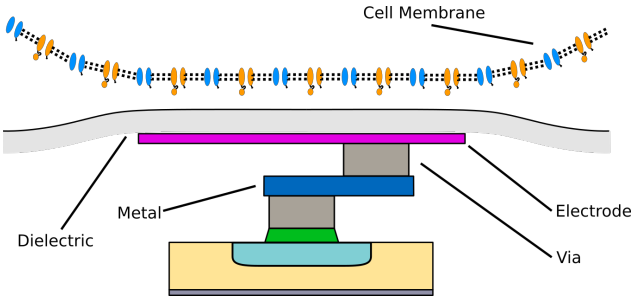


Figure 1.2:
Illustration of the operating principle of an EOSC. Charges on the electrode result in a non-faradaic current in the electrolyte. Current density is strongest in proximity to the dielectric and can be used to depolarize the neuron's membrane.

voltage range for stimulation [1].

The taken approach in current data acquisition systems for this class of devices is illustrated in Figure 1.3.

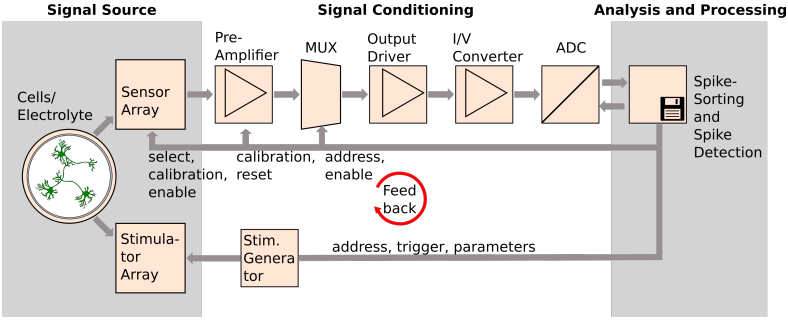


Figure 1.3:
Illustration of an in-vitro biohybrid system based on conventional computing architectures.

[1] R. Zeitler, Venneos personal communication

1.1.1 Typical Applications of MEA

These methods of investigation allow the graph-theoretical study of network formation [Feldt et al., 2011]. The large sensor area of $2.8 \times 3.8mm$ allows the investigation of both cultures and slices of brain tissue. The use of MEAs allows reliable data acquisition over long time-spans due to the extracellular method of measurement which does not damage the cells [Hutzler and Fromherz, 2004]. Due to the high resolution of this group of devices, assemblies of neurons can be stimulated and recorded from with high selectivity. The signals acquired in this way have amplitudes in the range of $300-800\mu V$. Figure 1.4 displays the signal acquisition capabilities of the technology with a hippocampal slice.

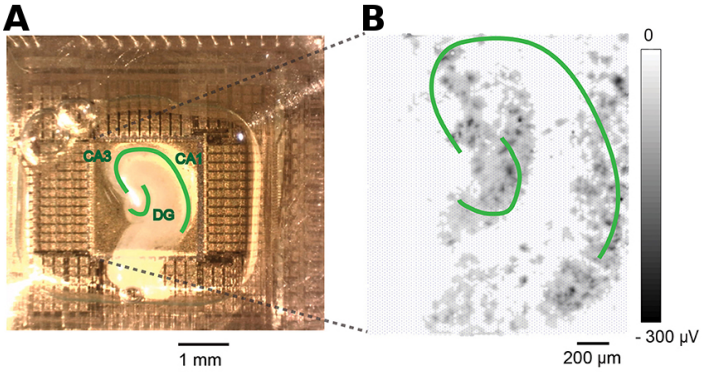


Figure 1.4:
A) MEA recording from a hippocampal slice, B) recorded activity in dentate gyrus slice [Gong et al., 2016]

Besides academic use for functional network analysis, in the currently emerging fields of application, monitoring of population activity is used for screening for neurotoxins like botulinumtoxin [Jenkinson et al., 2017].

1.1.2 Limiting Factors in MEA Based Biohybrid systems

Especially in neuroprosthetics, implantable MEA based biohybrids could find application, in for example the disruption of pathological activity in epileptic seizures. A second potential use is in long-term studies of network formation.

Several factors limit the usability of large electrode arrays in closed-loop interactions with populations of neurons:

- On a computational level, researchers face an enormous bandwidth challenge in the application of MEA over the course of time periods that are anything longer than seconds. With the following specifications taken from Lambacher et al. [2011], 128×128 sensor transistors are sampled at $6kHz$, resulting in
  $\sim$ 5 GigaSamples per minute. Assuming a $10bit$ accuracy, event these conservative figures result in $\sim$ 6.25 GigaBytes per minute.

- The involved computational steps for spike-detection and sorting are resource intensive and limit the number of channels which can be observed at a given time. Typically, this problem is approached by hand-selecting a subset of channels and reducing the sampling rate.

- If the computational resources of conventional hardware do not suffice to process data in an online fashion, data needs to be logged for later analysis. In network development studies that would collect recordings over days, at data rates of $\sim 6.25 \frac{GB}{minute}$, this results in the collection of terabytes of data so that even memory resources become a limiting factor.

Our aim is therefore to find a method which allows the application of neuromorphic hardware in the context of biohybrid systems. Here, the focus is on the creation of an interface, rather than the optimization of the neuromorphic hardware in itself.

1.2 Event-Based Communication

Compared to conventional signal acquisition with fixed sampling rates and sequential processing, the biological neuron processes data from its presynaptic information sources in a drastically different fashion. We propose a biologically inspired approach to signal processing to match the operating principles of the system under investigation to overcome bandwidth problems in biohybrid systems. Biological neurons' axon hillocks integrate over currents from the neurons dendritic tree and a stereotypical cascade of ion-channel activation is triggered, once the potential across the cell-membrane reaches a certain threshold. This cascade gives rise to a membrane depolarization and its subsequent reset to baseline. This signal is called the action potential. Conceptually, a neuron's spiking mechanism can be considered a delta encoding of the input current presented to it, where the output of the neuron is the information that the integral of the input signal surpassed a threshold. Here the action potential encodes the threshold-crossing event and propagates it through the neuron's axon. The compressive nature of this encoding is used in neuromorphic sensors, such as the Dynamic Vision Sensor (DVS) [Lichtsteiner et al., 2006] to create power-efficient devices with high temporal resolution and low data rates. A widely used protocol for the communication of action potentials in neuromorphic circuits is Address-Event Representation (Address-Event

Representation (AER)), a protocol for asynchronous communication of the senders'/receivers' IDs.

Event-based MEAs that are similar to the DVS in the way they represent the acquired electrophysiological data can interface directly with neuromorphic processors through a bidirectional stream of address events for both stimulation and recording.

Several groups are currently pursuing the development of such devices e.g. [Corradi and Indiveri, 2015a] and have demonstrated prototypes [Gupta et al., 2016].

Besides the reduction of data with this form of communication, the adjustment of the event-triggering threshold allows for a rudimentary spike detection in the electrophysiological data which reduces computational load in the analysis. Event-based interfacing with neuromorphic processors enables spiking neural networks for the use to classify the data presented [Qiao et al., 2015].

1.3 Towards Neuromorphic Biohybrid Systems

In contrast to conventional computing architectures, such as the ones employed in the current state-of-the-art analysis of electrophysiology data, neuromorphic hardware is particularly well suited for interfacing to biological neurons since neuromorphic processors are closely oriented towards biological operating principles, and perform computation in the same parallel, asynchronous and distributed fashion. In this way, neuronal activity from a source can be processed by target neuron circuits as it arises. The parallel nature of the neuromorphic processor implies that, in the case of a simultaneous firing

of biological neurons, the processing latency is less affected than in the sequential processing of every source, in the same scenario. Event-based MEAs also implement a rudimentary spike-detection mechanism that triggers the transmission of an address-event. This allows a reduction of pre-processing steps that would be necessary for a frame-based data acquisition with conventional computer architectures. The creation of a neuromorphic biohybrid system would promise an efficient parallel processing of data in biological real-time for the creation of a feedback signal that guides the stimulation protocol in a closed-loop scenario. However, even with the described advantages of neuromorphic processors in the given application, the large number of MEA channels provides a challenge to neuromorphic processors with limited numbers of synapses over which input to an emulated network can be provided. The neuromorphic engineering principle aims at reproducing biophysical processes in silicon to create bioinspired computing architectures, therefore, in search of an approach to deal with restrictions on the number of available synapses, a look at biological systems suggests itself. Here, sustaining functional synapses requires energy, so there is an incentive to keep the amount of interneuronal connections close to a minimum. The neuroscientific literature suggests that this resource optimization strategy is implemented through several strategies that limit the number of dysfunctional connections. These mechanisms can be summarized as activity-dependent structural plasticity. A system that employs structural plasticity mechanisms for the selection of a subset of sources from a large vector of potential event sources might look like that illustrated in Figure 1.5. Here, a digital co-processor is involved in the source selection process and the resulting stream of events is forwarded to the neuromorphic processor, which implements a form of classification on the input provided. The resulting

firing activity of the processor is routed to the stimulating electrodes of the MEA to form a closed loop.

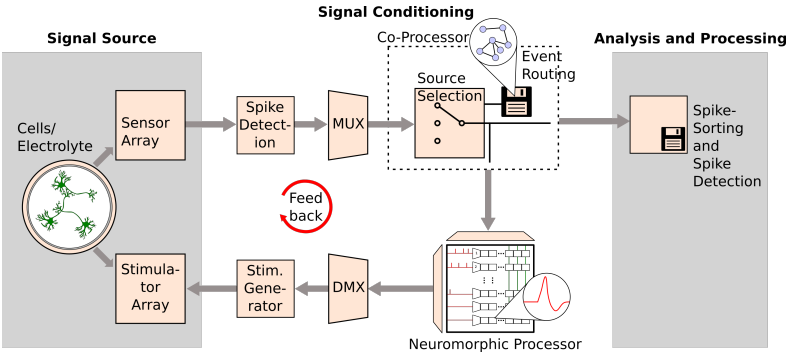


Figure 1.5:
Proposed neuromorphic implementation of a biohybrid system, centered around a neuromorphic processor. The focus in this thesis is on the digital co-processor and on the algorithms it implements for source selection

In the present work, we focus on the selection of sources to enable future work on neuromorphic biohybrid systems.

1.4 Structural Plasticity in Neuromorphic Systems

Employing structural plasticity for source selection when facing the problem of limited inputs to the neuromorphic system, extends the philosophy of neuromorphic engineering, by adding a degree of biological plausibility to the digital routing of input events to the neuromorphic Integrated Circuit (IC). Using algorithms that are inspired by activity dependent structural plasticity fits particularly with the use in event-

based MEAs since the biological network under investigation re-configures following the same principles while it is observed. This re-configuration, which includes the growth of neurites and the continuous generation and degeneration of synaptic connections occurs over the course of hours to days. As connectivity in the cell culture changes due to this developmental and activity dependent structural plasticity in vitro, and as environmental influences such as temperature and humidity affect the signal, electrode activity shows a variability over long time-scales that needs to be accounted for in the selection of channels. This process is performed through an activity dependent structural plasticity in silico. A source electrode selection mechanism that follows the same principles promises to allow the choice of those electrodes which maximally correlate with the firing activity of the postsynaptic neuromorphic processor which forms the artificial side. The aim of this work is to create a neuromorphic system capable of implementing a structural plasticity model for source selection. In addition, structural plasticity brings several other advantages, both in the neuromorphic and the conventional computational neuroscience context: Besides its use in the proposed selection of information sources, structural plasticity also has positive implications on network performance outside of neuromorphic engineering. Networks that model associative memory, are shown to be capable of storing several memories up to a maximal memory capacity in their weight matrix. Exceeding this capacity, corrupts all stored patterns, an effect described as Catastrophic Forgetting. Knoblauch et al. [2014] show that here, structural plasticity is successfully increasing memory capacity and prevents catastrophic corruption of the stored patterns. The argument here is that the presence and absence of a synapse is incorporated in the pattern and provides a further bit to encode information in the network, while not

functionally affecting the postsynaptic activity which would distort memory retrieval. Apart from the increase in memory capacity, a further advantage of structurally plastic networks is the limitation of the number of synapses. Since the number of synapses present per neuron is limited in neuromorphic systems, it is necessary for a structural plasticity rule to respect the system's upper limit of connectivity. Hence, the focus here is on the creation of a deterministic model in which it is easy to incorporate limits to graph measures such as the maximal fan-in of a neuron, where other authors use probabilistic methods (see [Fauth et al., 2015]). The fact that the ROLLS neuromorphic processor that forms the artificial side in our biohybrid performs the routing of events outside of the chip and makes no use of hierarchical routing architectures, was exploited by implementing a digital co-processor on an Field Programmable Gate Array (FPGA) which interacts with the ROLLS chip and the network's adjacency matrix in external memory. The advantage over an integrated solution implemented within a neuromorphic processor, is that by choosing configurable hardware as our computational substrate, the co-processor is kept as modular and flexible as possible. This design choice aids the explorative approach taken, in finding algorithms to incorporate activity dependent structural plasticity in a neuromorphic system [George et al., 2015a]. This is supported by the co-processors programmability in the *C* programming language.

1.5 Structure of the Thesis

In this thesis, we apply methods of activity dependent structural plasticity in a neuromorphic system for the selection of event sources. These event sources are formed by the electrodes of an event-based MEA.

The structure of the thesis follows the design process of the created system and can be divided into three main parts:

- In Chapter 2, mechanisms that contribute to activity dependent structural plasticity in biology are reviewed and abstracted into computational paradigms that are applicable in the digital co-processor designed in part three.
- In Chapter 3, the components of the neuromorphic system are described, and the design for a digital co-processor that allows the simulation of an structural plasticity algorithm is described at the level of hardware.
- In Chapter 4, the development of the structural plasticity algorithm itself and its operation in the target environment is presented along with an analysis of the implications of the functionality implemented on structural rearrangements within the network.
- The Chapter 5 considers use-cases and their verification in a more broad perspective. Here, the optimization process on conventional hardware is presented as well, to highlight the computational advantages gained from the addition of structural plasticity to spiking neural networks.
- A selection of publications that emerged from this project is presented in the appendix, together with background

information, that is referenced to throughout the thesis.

Each chapter included in these sections concludes with an overview that summarizes findings and lessons learned that informed the design decisions made in the subsequent chapters of the project. Approaches to overcome the shortcomings of the solutions presented are discussed in the discussion chapter. The conclusion chapter outlines the taken approach and promising directions of future research.

2

Principles of Synaptic and Structural Plasticity in Biological Neurons

This section of the thesis aims to review findings in neurobiology, with a focus on the extraction of functional principles that can contribute to the activity-dependent re-configuration of connectivity. In the organization of the nervous system, synaptic connections between neurons play a crucial role, both with respect to computation and to memory. Whereas intracellular signal propagation is implemented through electrochemical signaling, the means of information transmission across a synapse is biochemical, using neurotransmitters. These transmitters are stored presynaptically in vesicles, which dock to the presynaptic side of the synaptic cleft and release their content as a reaction to changes in membrane potential, as illustrated in Figure 2.1. On the postsynaptic side, transmitters are bound by receptors, that act in a metabotropic or ionotropic fashion. Ionotropic receptors affect the conductivity of postsynaptic ion-channels, which in turn cause an electrochemical effect that is propagated through the dendritic tree, towards the soma. Synaptic plasticity describes mechanisms which alter synaptic properties such as receptor conductivity, vesicle release probability, and protein synthesis, which

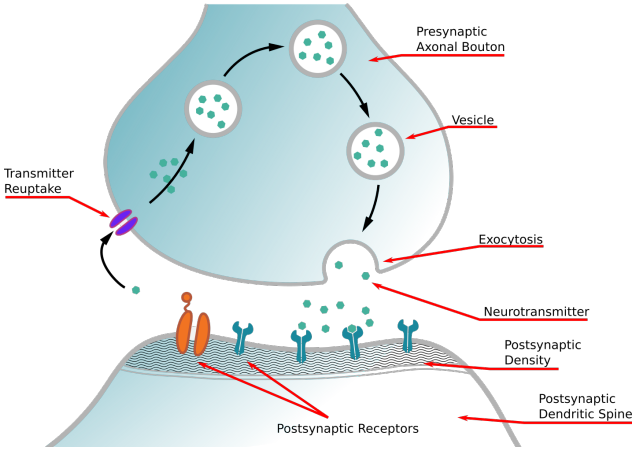


Figure 2.1:
Schematic illustration of synaptic signal transmission through neurotransmitters. Alternative reuptake mechanisms and receptors are not shown for reason of clarity

in turn affect the amplitude of the resulting excitatory or inhibitory dendritic current. In this respect, the synapse forms a memory element that stores the degree to which correlated activity occurred between the pre- and postsynaptic neuron in the past. Computationally, passive membrane properties and voltage-gated ion-channels implement the computational properties of the synapse and give rise to the characteristic shape of postsynaptic currents. In contrast to von Neumann architectures, computation and memory are co-located in the synapse. Over the course of maturation of an organism, and even during adulthood, new synapses are formed in a process called synaptogenesis, while others are destabilized and degenerate. This rearrangement of the neural network is called structural plasticity.

For the creation of a biologically inspired solution to the

source selection problem (outlined in the Chapter 1), the first step was to gain a deeper understanding of the mechanisms that underlie synaptogenesis in the mammalian brain. To do so, in the following, we review the body of literature on experimental findings in molecular biology and form hypotheses on the functionality of the observed interactions of ions and proteins. Moreover, we abstract these functions to their underlying computational primitives to reach a level of description which can be modeled in the target hardware framework. In particular, the focus of this review is on dendritic spine motility and postsynaptic plasticity. We conclude that synaptic plasticity (Changes in the synaptic efficacy) and morphological changes in the dendritic spine are strongly interlinked. Key mechanisms of this link that play an important role in structural plasticity are summarized at the end of the chapter. The approach taken differs from other attempts of modeling structural plasticity, that are more data-driven (see Butz et al. [2009]). Instead, the abstracted computational paradigms inform the algorithm design and optimization process.

As synaptic connections between axons and dendrites take up space in a limited three-dimensional volume, connectivity is limited by the available contacts between these neurites. Even in cases in which two neurons can form a potential connection, energy constraints limit the number of functional synapses in the network, as their maintenance requires a continuous uptake of energy. For these reasons, the biological system has a strong incentive to reduce connectivity to a minimum, which results in a network topology that can be represented by a sparse adjacency matrix. To sustain this sparsity, while maintaining high performance, the network continually adapts and re-arranges its connections to a certain extent.

This form of activity-dependent structural plasticity promises to yield gains in efficiency in the use of memory and resources

in a neuromorphic system. A suitable platform for implementing such reconfigurations through operations on the incoming and outgoing event-streams has been described in the previous chapter.

Analogous to the approach of starting from charge-carrier dynamics to develop neuron models, as used in neuromorphic analog Very Large Scale Integration (VLSI), our development of a model for activity-dependent structural plasticity begins from observations on synapse development and function. Here, we focus mainly on dendritic spine dynamics and the causes for morphological changes in individual synapses and their dendritic spines. This chapter highlights those mechanisms which are considered potential contributors to synaptogenesis in adulthood. The following chapter on implementation and design of the structural plasticity algorithm (Chapter 4) integrates these mechanisms to form an abstract model that can be used in the hardware platform to be created.

The focus in the development of a plasticity rule was on its applicability in neuromorphic systems, so simulations on a Personal Computer (PC) and the neuromorphic processor's design guided the choice of which computational operations were necessary to use structural plasticity as a means of resource optimization. These simulations and the final system are described in Chapter 5. In our approach, the biological literature informed the way specific functions were implemented, while computer simulations influenced the search for computational primitives in biology.

As works such as [Redondo and Morris, 2011] highlight, the conclusions about the computational primitives behind the biological processes observed to be involved in synaptogenesis remain partially hypothetical and would require verification to allow strong statements to be made about the biological system, as the exact biochemical foundations of structural plas-

ticity, i.e. in the dendritic spine, are subject to ongoing research. Instead, since the focus of this work was to create an application specific-solution, the interpretation of the processes observed remains subject to further research.

Section 2.1 provides the necessary background to understand the modeling choices made for the development of the Spike-Timing Dependent Plasticity (STDP) like algorithm presented in the subsequent chapter on the algorithm and its implementation. In Section 2.1, synaptic plasticity is discussed, and a link to growth dynamics in the dendritic spine is presented in the following Section 2.2. Activity dependent contributions to spine morphology and synaptogenesis are contrasted with the developmental aspects of network formation in Section 2.3. The computational implications of cell morphology and its passive membrane properties that give rise to effects such as synaptic co-activation are described in Section 2.4. This section also serves to contrast neuromorphic hardware with its biological counterpart. Where the previous overview of co-activation described a collaborative mechanism between synapses, Section 2.5 reviews findings on transport and scaffold mechanisms that may lead to competitive behavior.

A central concept captured to maintain the high parallelism of neuromorphic systems was that of computation on locally available information. The model later developed aims to operate only on those synapses that share the same postsynaptic neurons and only on events that are passed to these synapses to maintain a degree of biological plausibility. On the contrary, a more direct method of approaching structural reconfiguration could involve treating the adjacency matrix as a discrete two-dimensional plane, in which anisotropic filtering algorithms, developed for image processing, can be applied. This family of algorithms, however, operates on global information (the

complete adjacency matrix). This would impose challenges in an event-based framework, if structural reconfiguration is supposed to act on the same timescale as synaptic plasticity, between incoming events. The reason why this would be problematic is that the number of synapses to consider would include all present and potential synapses in the network. For the simulation based on conventional computing architectures, we have approached this problem by performing structural reconfigurations in a periodic fashion [Spiess et al., 2016], on a separate, slower, time-domain. This further departure from biological mechanisms was avoided in the neuromorphic implementation, through the restriction to computation on the postsynaptic neurons dendritic spines, which allows the use of a single time-domain.

2.1 The Role of Ionotropic Receptors in Synaptic and Structural Plasticity

In the following, we refer to the mechanisms of Long Term Potentiation (LTP), Long Term Depression (LTD) and short-term plasticity as *synaptic plasticity* that alter synaptic conductivity, in contrast to *structural plasticity* which includes morphological changes, here specifically on the dendritic spine. Synaptic plasticity, as an activity driven process of changing the synaptic efficacy, was first postulated by Donald Hebb in 1949 [Hebb, 1949]. In its earliest form, Hebbian plasticity describes the observation that the connection between two neurons is strengthened whenever one neuron drives the others spiking activity. The opposite case of the reduction of the strength of a connection, if the driving neuron's firing does not contribute to the firing of the postsynaptic neuron is also

necessary to prevent saturation to maximally strong connections and the resulting feedback loops. STDP extends this concept by introducing a dependence on timing into rules of synaptic plasticity to capture the degree to which a correlation between pre- and postsynaptic neurons fire in a more gradual fashion.

In STDP, the most commonly used function to describe the dependence of the synaptic weight on pre- and postsynaptic firing is an exponential fit to early electrophysiological measurements of synaptic plasticity by [Bi and Poo, 1998] (see Figure 2.2).
It has been stated, that various curves have been measured in different animal models and different regions of their nervous systems. In digital hardware, the weight update rule is usually evaluated numerically for the benefit of high precision in the weight update. Out of resource considerations, in the present work, a heavily discretized rule is implemented in the form of a look-up table (see Chapter 4).

The observed changes in synapse efficacy are triggered by the following signaling cascade. If activated by neurotransmitters, ionotropic receptors change the conductivity of an ion channel. In its activated state, the α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid receptor (AMPAr) is conductive to cations including Na^+ and Ca^{2+}. Upon the binding of glutamate, the opening of the channel causes an ionic current that leads to the depolarization of the membrane of the postsynaptic cell. The complex agglomeration of AMPAr and other receptors, bound by a scaffold of proteins, forms the functionally active zone of a synapse which is prominently visible in electron-microscopy of neuronal tissue. This structure is therefore called the postsynaptic density of a synapse. Its size is thought to serve as an indirect measure of the synaptic efficacy.

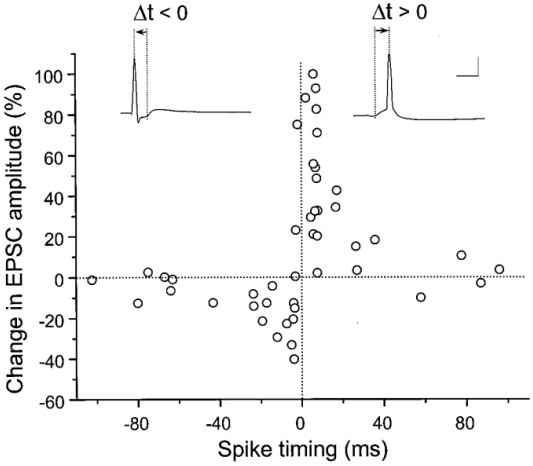


Figure 2.2:
Change in EPSC amplitude as a function of spike timing between the presynaptic and postsynaptic neuron ($t_{post} - t_{pre}$), taken from [Bi and Poo, 1998]

In the case of the N-methyl-D-aspartate receptor (NMDAr), change in conductivity of the ion channel is only achieved if depolarization of the postsynaptic spine leads to a repulsion of the charged NMDAr magnesium 'plug'. This mechanism makes N-methyl-D-aspartate (NMDA) a voltage-gated ion channel.

In an interplay between the two described receptors, AMPAr causes the depolarization necessary to activate NMDAr. Importantly, the depolarization works across synapses and even through back-propagating action potentials originating at the cells axon-hillock. Since NMDAr too is a glutamatergic ionotropic channel, this, in turn, leads to further influx of Ca^{2+}. If a postsynaptic depolarization is not present, the pos-

itively charged Mg^{2+} plug blocks the influx of Ca^{2+} through NMDAr. A further source of depolarization is formed by L-type voltage-gated calcium channel (VGCC)s that also activate on depolarization. Apart from depolarizing the postsynaptic neuron further as a means of signaling, the presence of an increased calcium concentration in the postsynaptic neuron triggers a CaMKII signaling cascade.

Once activated through calcium, the CaMKII enzyme is capable of phosphorylating target proteins. This activation is prolonged through autophosphorylation, even if the calcium signal has decayed. One of the primary mechanisms in synaptic plasticity is the phosphorylation of α-amino-3-hydroxy-5-methyl-4-isoxazolepropionic acid (AMPA) to increase the channel's conductance. As a secondary messaging pathway CaMKII plays a crucial role in long-term plasticity, through two mechanisms. Firstly, synapse-associated protein 97 (SAP-97) and Myosin-VI which are responsible for transporting AMPAr, are bound to the receptor, and the complex is moved into the membrane. This process is illustrated in Figure 2.3. Secondly, structural strengthening is enabled through the triggering of cytoskeleton development and the promotion of growth of the dendritic spine, contributing to structural plasticity. [Bonhoeffer and Engert, 1999] [Jourdain et al., 2003]

This relation of synaptic plasticity and changes in spine morphology is further underlined by the observed proportionality between spine volume and EPSC amplitude [Smith et al., 2003]

Using the outlined pathway, triggered by NMDA, new AMPA receptors are phosphorylated and enter the cell membrane in an activity-dependent fashion. Well known abstractions of this process are the spike-timing dependent models of synaptic plasticity.

Also, the synthesis and transport of new AMPA is performed in an activity dependent fashion, where transmem-

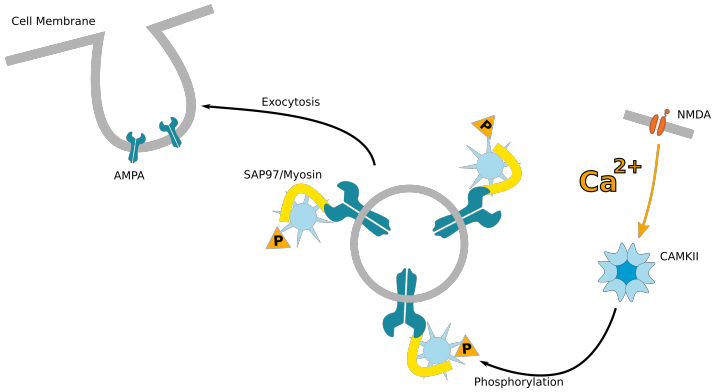


Figure 2.3:
The CaMKII signaling pathway promoting insertion of AMPA into the cell membrane

brane AMPA receptor regulatory proteins (TARPs) play a crucial role.

Likewise, LTD is related to the postsynaptic calcium concentration. Here, low levels of calcium lead to a loss of AMPA receptors in the active zone which may lead to a destabilization and shrinkage of the synapse as Butz et al. [2009] illustrate in their review on structural plasticity.

The here summarized observations lead us to the conclusion that it is possible to base a computationally efficient model of structural plasticity on a spike-timing dependent rule that evaluates a measure of spine morphology, analogous to existing weight update rules for the modeling of synaptic plasticity. This approach can be contrasted with an alternative of creating a stochastic process that determines synaptogenesis- and pruning events, and yields the ability to maintain a fixed number of synapses in an resource-constrained environment, such as a neuromorphic processor.

If more than one synapse is considered, NMDA's voltage gating implements a form of coincidence detection between neighbouring synapses. If closely neighbouring synapses are activated in a temporally correlated fashion, reciprocal depolarization causes an increased conductivity of the involved NMDAr. This mode of operation is especially important in the early stages of synapse formation, since developing synapses assume a 'silent' state, initially. At this stage, until $1-3h$ after synapse formation, no AMPAr are present in the postsynaptic density and the synapse does not directly contribute to the firing of the postsynaptic neuron. However, since NMDArs are expressed, a co-activation with a neighbouring, mature partner can occur. This leads to LTP through CaMKII and eventually to the transition into a fully functional synapse that also includes AMPAr. We hypothesize that through this mechanism, the maturation of synapses that are in the immediate neighbourhood of frequently and strongly activated mature neighbours is facilitated, whereas those newly forming synapses that see less depolarization by their neighbourhood, are restricted in their development. In this sense, the maturation of a silent synapse is here driven by correlated firing with a mature neighbour, rather than through the correlated firing of pre- and postsynaptic neuron. This is especially true in the distal part of the dendritic tree, where back-propagating action-potentials from the soma are attenuated due to the passive effects of the cell membrane described in Section 2.4. This co-activation may be contributing to the observed functional clustering of synapses. A more direct mechanism that could give rise to functional clustering is a causal link between LTP and synaptogenesis in the vicinity of the synapse that underwent potentiation [Lamprecht and LeDoux, 2004].

2.2 Activity-Dependent Morphological Changes

After highlighting the electrical mechanisms that are involved in the formation, the strengthening, depression and the destabilization of a synapse, the following section tries to capture key aspects of the more indirect biochemical mechanisms which support structural plasticity.

Studies that have investigated mouse barrel cortex show convincingly that spine and bouton growth dynamics are correlated with changes in the statistics of presented sensory input [Holtmaat and Svoboda, 2009]. Here, trimming a whisker and thus depriving a particular barrel in a rodent's somatosensory cortex of its predominant thalamic input, leads to an observable reconfiguration in the overall barrel cortex, in terms of connectivity. Areas that receive a reduced input activity respond by forming connections to those thalamic input sources which remain active, and connectivity to other cortical areas is also increased [Deger et al., 2016]. This finding was taken as a starting point to formulate a rule for structural reconfiguration that is driven by the firing activity of the neurons involved, through a mechanism which drives the motility of dendritic spines. Dendritic spines can be roughly categorized in their morphology as one of the following, illustrated in 2.4:

- **Filopodia** associated with a multitude of axonal partners in early development.
- **Thin and Stubby** Dendritic Spines that are associated with fewer partners and have enlarged postsynaptic densities that include receptors, which are fixed by scaffolding proteins.
- In later stages of maturation, spines become **mushroom shaped** or **cup shaped** and form the main site for glu-

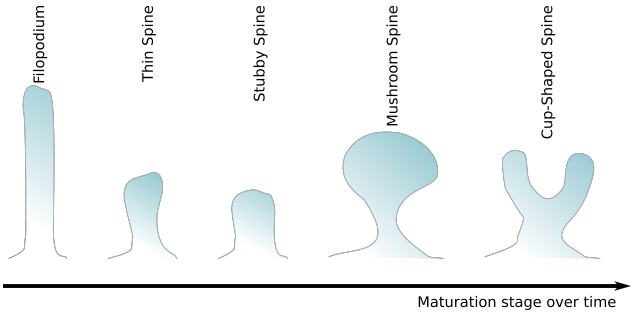


Figure 2.4:
Shapes of dendritic spines, following their development from left to right: Filopodium; Thin Spine; Stubby Spine; Mushroom Spine; Cup-Shaped Spine

tamate receptors. Interestingly, the spine's neck with its constriction forms a resistance that may cause large differences in membrane potential in the spines head, compared to more downstream compartments of the dendritic tree.

The taxonomy of spines presented here follows the observed developmental stages in their maturation. Although the change of morphology comes with electrochemical implications (e.g., calcium concentration and the resistance along the spines neck are affected by its shape), we distinguish only three types of spines in the later developed model, focusing on the main functional differences:

- Mature synapses that are subject to synaptic plasticity.
- Silent synapses that are only conductive, if co-activated with a mature neighbour.
- Potential synapses (locations where a contact between axon and dendrite is present but no functional synapse).

In the initial formation of a dendritic spine, [Yuste and Bonhoeffer, 2001] hypothesized LTP as the driving force behind maturation. This idea is rooted in the observation that after an initial delay no or little morphological modification is observable. Only when early LTP causes alterations in the synapse's efficacy, structural changes become observable too.

In a second stage, spines mature and grow in size with an increased number of receptors in the postsynaptic density [Yuste and Bonhoeffer, 2001]. The change from filipodic to mushroom-shaped causes a change in the capacity for calcium in the spine and therefore affects function. In this mature stage, dendrites are seen to bifurcate and show several postsynaptic densities. Moreover, new spines also form in the vicinity of those that undergo LTP.

As Redondo and Morris argue in their review on the synaptic tagging hypothesis [Redondo and Morris, 2011], the intracellular signaling pathways which cause changes in the morphology of spines and boutons can be distinguished from those that are responsible for synaptic plasticity. The authors assume that, on the presynaptic side, changes in synapse morphology are caused by the insertion of scaffold proteins and changes in the cytoskeleton that affect the ability of neurotransmitter vesicles to bind to the cell membrane and release the transmitter into the synaptic cleft. The ability for AMPAr to be anchored on the postsynaptic site is affected in a similar fashion. Through these mechanisms, which allow for long-lasting changes in plasticity, the foundation for changes in the synapse's efficacy is given, as the described morphological changes cause modifications, driven through spike-timing dependent plasticity.

2.3 Developmental Aspects of Topology Reconfiguration and the Wnt-Protein Signaling Pathway

Besides the previously outlined activity driven processes of synapse stabilization and destabilization in the adult network, especially during maturation of the network, activity independent mechanisms influence synapse formation to create an initial network. Members of the Wingless-related integration site (Wnt) protein family, for instance, are expressed during the growth of the dendrite. These promote synapse formation in hippocampus, also in adult networks. Moreover, dendritic arborization is promoted by the Wnt signaling pathway, which is confirmed by a targeted blocking, that leads to a decrease of complexity in the dendrite's morphology. [Rosso and Inestrosa, 2013]. Furthermore, a modulation of NMDA conductivity has been shown by Cerpa et al. [2011] and others.

The role of Wnt in developmental synaptogenesis inspired the assumption that the number of dendritic spines is proportional to the neuron's arborization and the total length of its dendrites. This assumption is later used to form a basic model of developmental plasticity, however it was discarded in the implementation of activity dependent structural plasticity in the neuromorphic system, due to the lack of a notion of morphology here. Moreover, changes in the cell-morphology during development may also have a secondary electrochemical effect on synaptogenesis by influencing the Ca^{2+} pathway. Arguably, this would imply that activity-dependent and developmental effects on synaptogenesis are not entirely separable. We did not follow this hypothesis further at this stage due to the limitations of the target system pointed out, however sim-

ulations on conventional hardware may profit from this notion in future work.

2.4 Cell Morphology and Dendritic Computation

A fact that has seen little recognition in the literature on spiking neural networks is that passive membrane properties form a link between cell morphology and electrochemical properties of neurons. The most commonly used models of spiking neurons, describe neurons as point-processes without spatial dimensions or morphology. Future generations of neuromorphic processors may profit from a compartmentalization of their dendritic input, not only for the sake of an increased biological plausibility but also with respect to computation. Mechanisms which cause local depolarization, for instance, may be used for heterosynaptic plasticity rules. Axonal delays could also prove beneficial in the processing of data with a temporal dimension, such as data relating to the motion of objects in a scene or acoustic input.

The neuron's morphology plays a central role in synaptic plasticity through affecting the shape of the postsynaptic depolarization, as the passive propagation of ionic currents along the neurite, usually described by cable equations, is highly dependent on the radii and lengths of the given compartments of the neurite.

The source of this postsynaptic depolarization can be either back-propagating action potentials from the soma or calcium influx through neighbouring synapses. The membrane potential's evolution over space (the length of the compartment l) and over time t, can be described by the following partial dif-

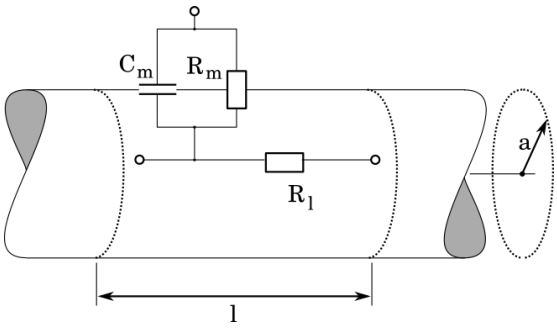


Figure 2.5:
Passive electrical properties of the cell give rise to the spatiotemporal voltage dynamics, described by the cable equation.

ferential equation,

$$\lambda^2(\frac{\delta^2 v}{\delta^2 x}) = \tau_m(\frac{\delta v}{\delta t}) + v$$

in which $\tau_m = c_m \times r_m$ represents the time constant of the membrane which represents an RC low-pass filter. The space constant $\lambda^2 = \frac{r_m}{r_l}$ captures spatial attenuation due to the resistances length.

The cellular membrane both induces capacitive and resistive elements that shape the voltage response to an injected current as outlined in Figure 2.5.

The involved passive elements are defined by the circumference of the membrane $r_m = \frac{R_m}{2\pi a}$; $c_m = C_m 2\pi a$ and the length of the compartment, affecting the intracellular resistance $r_L = \frac{R_l}{\pi a^2}$.

Compared to this dynamic, no similar location-specific effects are implemented in the neuromorphic processor. The membrane capacitance is implemented as a single capacitor which integrates over synaptic currents. Moreover, R_m is ne-

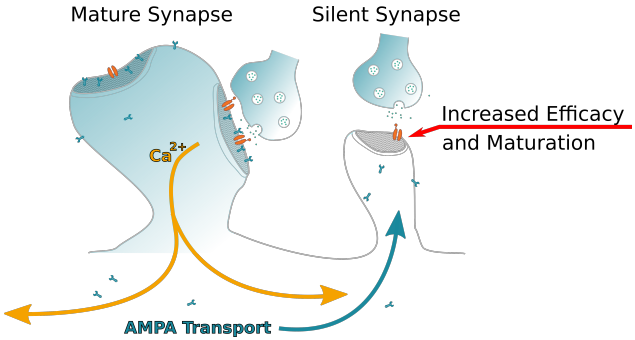


Figure 2.6:
Synaptic co-activation as a cooperative process that progresses the maturation of a silent synapse (right) if its activation coincides with the activation of a mature neighbour (left) or a backpropagating action-potential that causes depolarization. This change in membrane potential on the silent synapse removes the NMDA magnesium plug and leads to Ca^{2+} *influx. Following the* Ca^{2+} *signaling, AMPA is transported to the synapse and phosphorylated. This results in an increase in the synaptic efficacy.*

glected so that the resulting emulation of neuronal behavior can be described as a point process. In the case of the short-term plasticity synapses, postsynaptic currents are modeled as first order linear low-pass filters on an incoming digital pulse, to closely approximate the shape of an EPSC/Inhibitory Post-Synaptic Current (IPSC).

In this synapse model, the conductivity of the receptor channels is not voltage-gated, so these circuits perform like AMPA channels in the biological correlate. The absence of voltage gating and moreover the absence of any form of spatial dependency on the postsynaptic potential itself drove the decision to implement an STDP-like plasticity rule as part of the developed algorithmic model, executed on the co-processor on the FPGA.

Compared to modeling the processes which give rise to synaptic plasticity on an ion-channel basis, the approach of modeling LTP and LTD directly presents a shortcut that saves computational resources.

The downside is that the nonlinear nature of dendritic computation and backpropagating action potentials are not captured, and this reduces the comparability of the measured experimental results with biology, where the cell's morphology can potentially influence activity-dependent plasticity indirectly, through the mechanisms described.

Using the neuromorphic processor beyond its intended point-neuron desgin, to introduce the notion of cell morphology into the ROLLS processor, for instance by using the neuromorphic neurons as abstract descriptions of the nonlinear computation occurring in single dendrites and then combine them together to form compartmentalized 'super-neurons'. In the present work, this approach was not followed so as to not trade off the already limited number of available neurons for a (disputable) increase in biological plausibility.

Since no additions are made to the point-neuron model implemented in the ROLLS processor, dendritic spines can be considered neighbouring each other directly, not displaying cable effects between synapses and lacking mutual depolarization.

Within the framework of a point-neuron, the most intuitive approach to synaptic co-activation is the assumption that every dendritic spine is capable of affecting all of their neighbours in an equal manner. This approach makes better use of the hardware at hand than a more detailed model, which could, for instance, encode morphological effects in the propagation delay between the pre and postsynaptic side.

The mechanisms described result in a local computation specific to the dendritic branch, due to local depolarizations that travel downstream towards the soma. This effect results in a non-linear interaction of neighbouring synapses which, if temporally correlated in their activation, affect each other. Where this effect hypothetically leads to a binding of correlated inputs, the situation of mature synapses with two fully functional synapses looks very different. Here, co-activation means that the driving force for neighbouring synapses is reduced, which may lead to spatial separation since co-activation would cause a smaller EPSC than the summed individual activation, a fact neglected in the proposed system since the neuromorphic implementation is not conductance based but directly current based. In the proposed system, lateral inhibition through a population of separate inhibitory synapses that do not participate in the plastic reconfiguration aims at introducing a comparable effect. [London and Häusser, 2005]

A further local computation observed in biological neurons is that of shunting inhibition. As Liu [2004] demonstrates, inhibition distant from an excitatory activation on the dendritic tree tends to be summed linearly at the cell's soma. However, a

spatially close inhibition can produce highly non-linear effects on the dendritic potential, which are specific to a branch and do not act globally. This is another aspect that has not been considered in the later formulation of the model. In order to obtain local inhibition, one could affect the transmission probability of the presynaptic event to the postsynaptic destination depending on the presence of temporally correlated inhibition at the stage of the external processor. This approach was not taken due to memory considerations. Every postsynaptic neuron would need to maintain a memory of when the last inhibition occurred. Furthermore, every postsynaptic side of a synapse would need information of its position relative to any inhibitory synapse on the same postsynaptic neuron as a form of weighting the impact of inhibitory activation.

2.5 Scaffold and Carrier Proteins

Receptor synthesis takes place remote from the postsynaptic densities on the dendritic spines where plasticity mechanisms induce structural changes. AMPAr are formed in the endoplasmatic reticulum, where their synthesis and their exit are mediated by a group of regulatory proteins.

Although even distal dendrites contain endoplasmatic reticulum and ribosomes and are supplied with mRNA from the soma, the question is, how do proteins that participate in synaptogenesis and in LTP and LTD make their way through the dendritic tree to the particular synapse where they are required, as signaled through e.g. CaMKII.

Proteins, and also mRNA for local synthesis are transported through both active and passive mechanisms.

The role of active transport is taken over by motor-proteins that are members of the dynein, kinesin and myosin fami-

lies. This form of transport is typically mediated along microtubules and other cytoskeletal polymers, which allow direction specific transport.

Since location specificity of e.g. AMPAr transport and exocytosis is disputed, we follow the assumption here, that it is not specific at the resolution of single dendritic spines.

Passive transport takes place through lateral diffusion along the cell membrane. The movement dynamics of receptors and other proteins on the cell membrane can be described as Brownian motion.

Once the cargo reaches the synapse, assembly into complexes integrates receptors for example, into the postsynaptic density. In this process, scaffold proteins play an essential role in stabilizing and anchoring proteins to facilitate the LTP process. The role of forming clusters of glutamatergic receptors in the postsynaptic density is also fulfilled for example by postsynaptic density protein 95 (PSD-95) that itself is anchored to actin through other proteins [Bats et al., 2007]. PSD-95 has been shown to diffuse and re-distribute between synapses. Furthermore, its lifetime is shorter than that of the dendritic spine which may explain the seemingly random nature of synaptogenesis and pruning, as the scaffold is not static. PSD-95 fulfills a second role in trans-synaptic signaling, causing structural changes in the presynaptic site and affecting the vesicle release probability [Futai et al., 2007]. Also presynaptic scaffold proteins play an important role in the functioning of the synapse. In Staras et al. [2010] the authors observe a strong vesicle mobility that results in the depletion of neurotransmitters on neighbouring axonal boutons if a synapse is activated frequently and attribute this effect to a less pronounced scaffold.

Meyer et al. [2014] show that synapses that lack PSD-95 expression in the first hours after formation, are not capa-

ble of stabilizing. The binding of PSD-95 to the postsynaptic density occurs in an activity-dependent manner through the Ca^{2+} signaling pathway. We take this as an indication that synapses which lack PSD-95 are generally unstable. This hypothesis allows us to form a link between synapse activation and dendritic spine stability. Together with the observation, that insufficient scaffold implies a lack of AMPAr in the postsynaptic density, this allows us to formulate a model where a single activity dependent variable determines pruning and scilencing of a connection.

As Chistiakova et al. [2015] illustrate, homosynaptic plasticity alone, would lead to saturation if the network is continuously exposed to input and is continuously plastic. A straightforward mechanism to counter this problem is a homeostatic weight normalization which introduces a competitive mechanism that sustains a constant sum of weights, either in the local neighbourhood of a dendrite, over the entire set of synapses of a postsynaptic neuron [Royer and Paré, 2003], or within a population of neurons.

In biology, a great number of homeostatic mechanisms perform this adjustment of weights on different time-scales. For instance, AMPAr are not tightly anchored but remain flexible on the cell membrane and free to diffuse if not bound by scaffold proteins. This is especially true if the AMPAr are integrated into the cell membrane and not transported by active transporter proteins. Maas et al. [Kappel et al., 2015] argue that this effect could account for the noise in the efficacy of the synapse as described by [Marder, 2011].

Taking the passive means of transport and imperfect scaffolding into account, a local competition between synapses for receptors seems plausible for the implementation of a competitive mechanism, where resources tend to be collected by those synapses that receive maximal potentiation. Other synapses

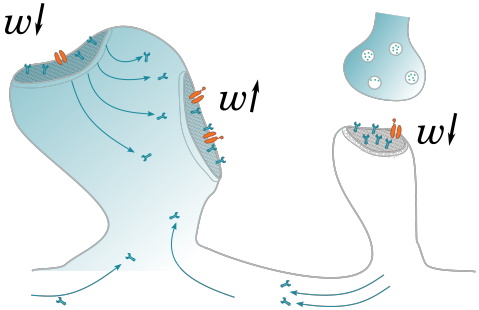


Figure 2.7:
Competition over receptors as a mechanism of subtractive normalization

may lose resources due to a weaker scaffold so that as a net effect, a dendritic compartment can be considered to contain a local pool of available receptors. This view is supported by the observation that in close neighbourhoods, dendritic spines' growth was negatively correlated [Govindarajan et al., 2011].

This pool of resources like receptors, would be supplied as new receptors enter the cell membrane external to the post-synaptic density, as a result of activity dependent synthesis. This form of long-term plasticity dependent on protein synthesis occurs three hours or more after plasticity induction.

Under the assumption that the Ca^{2+} signaling pathway (and with it long-term potentiation and depression), operates on a slower timescale than the diffusion of receptors in the local pool, the hypothesized competitive mechanism would imply that the maximum number of receptors present in the dendrite stays constant over short timescales. Competition between potentiating synapses for the receptors unbound at the loss of weaker synapses may form a type of weight normalization that (averaged over time) has a homeostatic effect on the neurons

firing frequency, as Butz et al. [Butz et al., 2009] argue.

In the biological reality, homeostatic effects such as the resource competition described above, do not take an immediate effect but occur in the hours after homosynaptic plasticity. This fact was ignored in the attempt to implement the mechanism. Besides the postsynaptic competition over resources, which we identify here as a means to counter the collaborative co-activation of silent and mature synapses, certainly effects on a microcircuit level, such as winner-take-all motifs that rely on inhibitory interneurons could implement a similar behavior at the cost of additional computational resources (i.e. neurons) in hardware. Due to the lack of a physical dimension and thus a lack of cell morphology in the present neuromorphic hardware, an abstraction of the described mechanisms needs to be made. In the algorithmic implementation, mechanisms of active transport and CaMKII's effect on the pool-size were neglected. Instead, weight homeostasis was implemented under the assumption of a fixed pool of AMPAr that does not change in its capacity over time. The quantity of receptors present is initialized as $N \cdot w_{init}$ and is equally distributed over all synapses present $n_i \in N$ (with N being the set of present synapses). If the implemented weight resolution allowed real-numbered values, potentiation of one weight would cause depression in all others. The weights used have a limited resolution so that expressing the depression of neighbours by $\frac{\Delta w_{ij}}{N-1}$ would lead to rounding errors. As an alternative approach, that moreover saves clock-cycles, we reduce the $n_{1..s}$ smallest weights on the postsynaptic neuron, each by a value of one. Here the number of synapses $\mid s \mid$ that are depressed, is $\mid s \mid = \Delta w_{ij}$. Similarly, the depression of a neuron (a negative weight update Δw_{ij}) causes the potentiation of the $\mid s \mid$ largest post-synaptic weights (see Figure 2.7). In the implementation targeting neuromorphic hardware, where a fixed number of

available synapses is one of the properties of the system, and no cell-morphology exists, all synapses are considered direct neighbours, so that instead of acting in the local neighbourhood, the pool described comprises the entire dendritic tree and the implied weight homeostasis is performed over all input weights of the neuron as a means of weight normalization.

The presented mechanisms exclude the role of astrocytes in structural plasticity, although there is strong evidence of astrocytic contribution to synaptogenesis, and pruning [Chung et al., 2015]. Thrombospondins, synaptogenic factors that are expressed during the first postnatal weeks, as well as after injury affect functional connectivity, by forming a signaling pathway that interacts with the calcium channel subunit Cacna 2d1 on the neuron. Further evidence shows, interaction with the postsynaptic adhesion protein neuroligin 1, that causes the increase of synapse numbers in hippocampal neurons in vitro. Importantly, also presynaptic effects on the vesicle release probability guided by astrocytic Cholesterol release, and postsynaptic influences on AMPAr are observed, that highlight astrocytes role in synaptic- and structural plasticity. Based on these findings, alternate hypotheses for the above mentioned mechanisms that implement substractive weight normalization, as well as pruning- and synaptogenesis could be formulated to give a more detailed model of the involvement of glial cells. This approach was not followed however, since the primary aim in the present work, was the creation of an computationally efficient model that optimizes resources for the neuron-centric hardware at hand.

2.6 Summary

The goal of the present review was the identification of mechanisms of synaptic plasticity and dendritic spine dynamics in networks of biological neurons. For this analysis, we reviewed the relevant literature and abstracted the reported findings into computational primitives. In the following chapters, these primitives are made use of in the development of algorithms for activity-dependent structural plasticity, driven through a Hebbian-like learning process. Here, a translation of principles from the framework of biological networks to neuromorphic networks was performed, with the consideration of differences in environment and design in mind.

Table 2.1 summarizes the identified mechanisms that are observed to have an impact on dendritic spine morphology, synaptogenesis, and pruning of synapses. For the goal at hand, we largely excluded trans-synaptic signaling mechanisms and the presynaptic effects on axonal boutons from the review since their implementation in a scenario with external presynaptic event sources would be complex and resource intensive. The only exception here is the introduction of a maximum d_s of postsynaptic destinations that a source is allowed to connect to. Since the focus of interest for us was the study of mechanisms that occur in dissociated cell cultures that underwent centrifuging to reduce the number of glial cells, the role of astrocytes was excluded from this review of structural and synaptic plasticity. Focusing on the mechanisms described allowed us to gain deeper insights into the type of cultures that form the subject of future work in MEA based biohybrids. In the following chapter, the implementation of the computational primitives that we abstracted from biological observations is detailed.

Biological Observation	Resulting Effect	Computational Primitive
Weak scaffold, unspecific exocytosis of AMPA, Brownian motion in the membrane. See Section 2.5	Subtractive weight homeostasis.	Competition for the resources such as receptors.
PSD-95 diffuses, has a short live time. See Section 2.5	Instability of silent synapses	Time-window for destabilization.
Calcium dependent dendritic spine dynamics. See Section 2.2 and 2.1	Synaptic plasticity is correlated with structural plasticity.	Structural plasticity as boundary condition for synaptic plasticity.
Cytosceleton stability is activity dependent. See Section 2.1	Activity dependent pruning.	Weight dependent pruning.
Heterosynaptic co-activation through local depolarization. See Section 2.4	Maturation of silent synapses.	Local collaboration of synapses.
Passive membrane properties result in strong local co-activation. See Section 2.4	Functional clustering of synapses.	Instantiation close to strong neighbours [Kleindienst et al., 2011].
Presynaptic vesicle pool is shared among a local neighbourhood. See Section 2.5	Local presynaptic competition.	Limited fan-out of the presynaptic neuron.
Wnt and others control branching. See Section 2.3	Number of dendritic spines is proportional to a neuron's arborization.	Distance to soma can be used as an estimator for synapse density.
Synapsin is distributed uniformly, branching occurs distant to soma (Maschietto, personal correspondence). See Appendix A.7	Synapse density increases as a function of distance to the soma, as neurites branch.	Distance to soma can be used as an estimator for synapse density.

Table 2.1:
Observed biological mechanisms and the interpreted computational primitives which inform the algorithm design in the following chapters

3

The Neuromorphic Processor with a Digital Co-Processor

The neuromorphic processor, on which we want to perform structural reconfiguration of connectivity, provides an array of unconnected neuron circuits, which communicate with an external router, which implements the topology of the network. This router takes topological information of the network's structure from an adjacency matrix in external memory. To create a suitable platform for the simulation of structural plasticity in a neuromorphic context, we created a digital co-processor on an FPGA, which can access the topology of the network and modify it through read and write operations on the external memory. By monitoring incoming presynaptic activity, as well as outgoing stimuli to target neurons, the co-processor is capable of executing a plasticity rule that determines the structural strength and maturation stage of the connection from the presynaptic to the postsynaptic neuron. In the following, we first describe the overall neuromorphic system and its components and subsequently focus on the description of the digital co-processor and the surrounding programmable System-on-Chip (pSoC). We also present a validation of the design applied to a toy-problem.

The structure of the neuromorphic system is layered in several hierarchies that are aimed at enabling fast and flexible development.

At the lowest level, the ROLLS processor performs the bulk of the computation involved in the emulation of spiking neural networks in the analog domain. The intermediate level is formed by a pSoC implemented within an FPGA. The highest level of processing is performed on the host PC optionally included in the system for forming interfaces to remote event sources via the User Datagram Protocol (UDP) and for visualization and post-processing, as well as event-generation and preprocessing of conventional data streams. Figure 3.1 illustrates the different layers of the system.

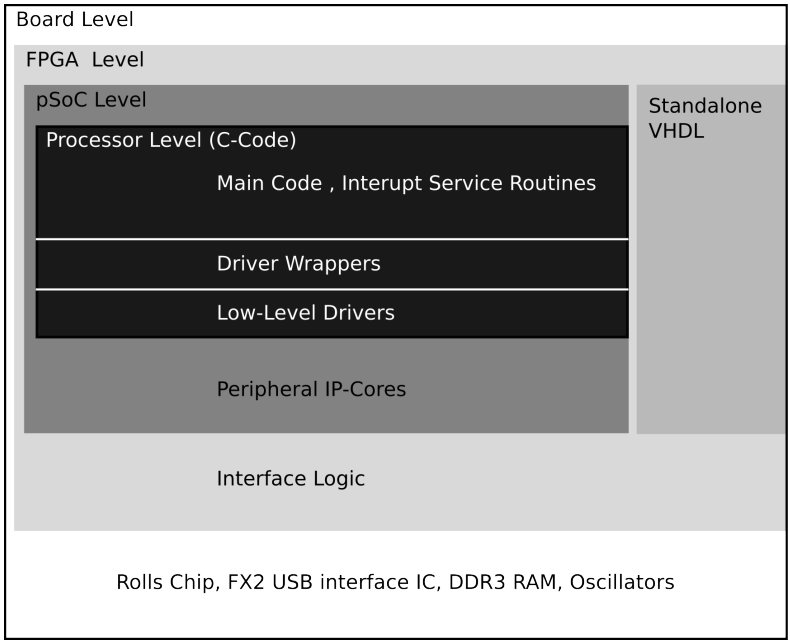


Figure 3.1: *Hierarchical organization of the created system*

This chapter mainly focuses on the description of the digital pSoC and its interaction with the main component of the sys-

tem, the neuromorphic processor. In this context, the pSoC fulfills several roles:

- Setting biases to tune the neuromorphic processor's parameters
- Establishing topology by looking up event destinations in the adjacency matrix in external memory
- Performing adaptive reconfiguration of topology
- Providing interfaces to external inputs via Universal Serial Bus (USB)
- Execution of complex stimulation protocols with precise timing (see appendix A.3)

The pSoC interfaces to the host computer providing a bridge to interfacing partners via UDP by translating UDP Packets to USB data and vice-versa. Data is made available to the pSoC via USB by use of a Cypress Fx2 IC that communicates with the FPGA through a First-In-First-Out (FiFo) interface for development purposes, and a serial RS232 connection allows the transmission of diagnostic data to the host PC. The firmware of the Fx2 and the corresponding drivers were provided by Daniel Fasnacht in previous work but customized to provide a user-friendly event-transmission based on single words of $32bit$ width that can hold identifiers, data, and time-stamps.

In the following, the different components of the neuromorphic system centered around the ROLLS chip are described, and the communication protocol involved is outlined step by step, following the flow of information from the neuromorphic processor to the host computer.

3.1 Analog Sub-Threshold Neuromorphic Processors

When Carver Mead realized in the 1980s that both ion dynamics across the cell membrane and carrier dynamics in transistors operating in the sub-threshold domain are following Boltzmann statistics, a new design paradigm entered the field of CMOS circuit design.

The underlying physical mechanisms of charge carrier transport that characterize sub-threshold transistor operation (gate-source voltage $V_{gs} < 0.7V$) is diffusion. If the transistor is operating in saturation (the voltage between source and drain terminals V_{ds} is larger than $4 \times U_t$), this diffusion can be assumed to be unidirectional to give rise to a current $I_{ds} = I_o e^{\frac{\kappa V_g - V_s}{U_T}}$ across the transistor's channel. Interestingly, this behavior resembles those processes that are observed in the transport of ions such as potassium, sodium, and chloride that diffuse across the biological neuron's membrane.

In both cases, diffusion gives rise to an exponential relationship of current to trans-membrane or source-drain voltage.

The realization that biological systems and subthreshold CMOS transistors in saturation share the same Boltzmann statistics lead to the emergence of neuromorphic engineering as a school of thought in analog circuit design, which aims to reproduce the behavior of biological neurons and synapses by operation in the subthreshold regime, to form information processing architectures that are energy-efficient and biologically inspired.

Subthreshold analog VLSI is well suited for the implementation of neuron models in hardware for two reasons. Firstly, currents between the source and drain terminals in this operation domain are in the nano- to femto-Ampere range, which

results in drastically reduced power consumption compared to conventional digital electronics.

Secondly, the emulation of neuronal behavior also profits from the nonlinear transistor properties that allow a straightforward biologically plausible implementation of neuron models with a small number of circuit elements. Neuron models such as the adaptive exponential leaky integrate and fire model by [Brette and Gerstner, 2005], are capable of displaying a rich set of physiologically plausible behaviors in their description of the neuron's membrane voltage as a function of input current and time. This family of models is suitable for implementation in subthreshold VLSI due to the small count of circuit elements needed to emulate the model's differential equations.

Similar to how biological neurons perform computation in a local and analog fashion and transmit information digitally through stereotypic action potentials, the action potentials of neuromorphic neurons are propagated through the network of circuits as 'spikes'. In both cases, synaptic plasticity is used to let the system learn and adapt.

The state-of-the-art approach to modeling neurons in analog VLSI is illustrated in Figure 3.2, which points out the differences in morphology between biological and neuromorphic neural networks.

The ability to emulate spiking neural networks in an asynchronous and parallel fashion opens several fields of applications in autonomous robotics and distributed sensor networks [Galluppi et al., 2014] [Milde et al., 2017]. Furthermore, neuromorphic processors are capable of interacting seamlessly with event-based sensors such as the dynamic vision sensor (DVS) [Lichtsteiner et al., 2006] and dynamic audio sensor (DAS) [Liu et al., 2010]. Here, an AER stream of 'spikes' can be processed by spiking networks for feature-extraction and classification tasks in hardware. [Corradi and Indiveri,

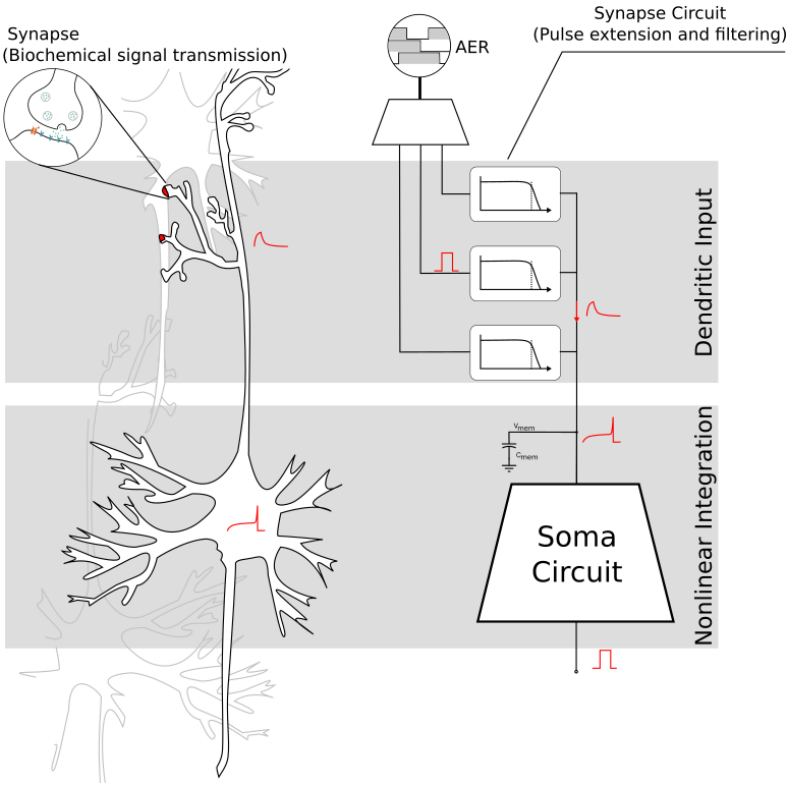


Figure 3.2:
Abstract comparison of a biological neuron with its neuromorphic emulation. Nonlinear integration is performed by the neuron's axon hillock and its corresponding circuit equivalent. This component receives input EPSCs and IPSCs from the synapses that connect to it via the dendrite. In the case of the neuromorphic emulation, inputs to the synapses, as well as the neuron's output action potential are digital and encoded in the AER format.

2015a] [Boi et al., 2016] [Osswald et al., 2017]

Where the underlying characteristics allow an emulation,

rather than a simulation of neuronal behavior in silicon, the closeness to biological systems also imposes comparable computation
-limiting constraints. These constraints include the vulnerability to thermal noise and spatial mismatch in the layout of the circuit due to manufacturing tolerances and therefore have an effect on the performance of the circuit.

Neuromorphic engineers argue that the choice of designing systems that are constrained in ways that are similar to biological systems leads to solutions that are more biologically plausible than abstract simulations and thus can provide insights in computational neuroscience while featuring the energy efficiency and fault-tolerance observed in biological systems.

A deviation from this philosophy is seen in the approach to creating architectures of large numbers of interconnected neuron circuits. Given the planar technology in IC design, it is impossible to provide the same degree of connectivity of axons and dendrites that three-dimensional biological population of neurons feature if the design resembles direct one-to-one connections between neuron circuits. Furthermore, the growth processes and dynamic re-arrangements of network topology observed in biology, are not implementable in hardware that is static after the point of fabrication.

3.1.1 The ROLLS Neuromorphic Processor

The ROLLS, used in this thesis, is a neuromorphic full-custom mixed-signal VLSI device with learning circuits that emulate the biophysics of real spiking neurons, fabricated using a standard 6-metal $180nm$ CMOS process [Qiao et al., 2015]. It occupies an area of $51.4mm^2$ and has approximately 12.2 *million* transistors.

This chip forms the core of the neuromorphic system which

was created for developing algorithms that implement a form of structural plasticity.

The ROLLS emulates 256 neurons that receive input from a total of 133,120 synapses of which 256×256 are learning synapses and 256×256 programmable synapses. Both arrays of programmable Short-Term Plasticity (STP) and of Long-Term Plasticity (LTP) synapse circuits are implemented. Both synapse arrays consist of analog circuits, that can reproduce biophysically realistic (short-term and long-term) synaptic dynamics, as well as digital circuits that can set and change registers which control for example network configuration settings or programmable weights (see [Qiao et al., 2015] for a thorough description and characterization of these circuits). A chip micrograph of the fabricated device is shown in Figure 3.3

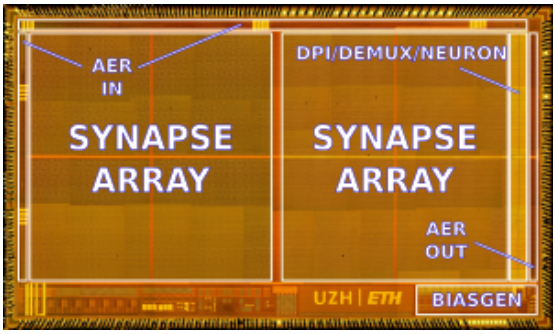


Figure 3.3:
Micrograph of the ROLLS processor silicon die, illustrating the ratio of synapses to neurons and the respective chip real-estate necessary

The silicon neuron circuits implement a model of the adaptive exponential integrate-and-fire neuron [Livi and Indiveri, 2009] that has been shown to reproduce electrophysiological recordings of real neurons [Rossant et al., 2010] [Brette and Gerstner, 2005] accurately. All synaptic currents are inte-

grated by low-power log-domain pulse integrator filters [Bartolozzi and Indiveri, 2007] that can reproduce synaptic dynamics with biologically plausible time-constants. An in-depth description and evaluation of the synapse circuits used can be found in the Appendix (See A.3) Action potentials generated by the neuron circuits are detected by asynchronous digital circuits on the chip that transmit address-events to the outside. Address-Events received by the chip result in the stimulation of neurons, or are used for reconfiguration of a synapse or neuron (depending on the address). Model parameters for manipulating both synapse and neuron circuit behavior are applied as bias-voltages. These voltages are created by an on-chip digital-to-analog converter, which is interfaced through a custom serial protocol [Delbruck et al., 2010]. Peripheral asynchronous Input/Output (I/O) logic circuits are used for receiving input spikes and transmitting output spikes, using the AER communication protocol.

In order to create arbitrary network topologies with large numbers of neurons, the design choices made, balance biological plausibility with scalability in the target CMOS environment. The foundation of these approaches is the use of bus communication, where shared communication lines replace axon to dendrite connections. In this framework, the direct communication of a neuron's action potential is replaced by the transmission of its identity to encode a spike 'event'. Any of N neurons transmit their identity $n \in N$, so that the number of neurons determines the width of the bus $d_{AER} = \lceil \log_2(N) \rceil$. Because of the asynchronous nature of communication in the absence of a global clock, a handshake protocol negotiates data transfer with the receiver. In the described AER of action potentials, this handshake is four-phased, using a request signal, driven from the transmitter side and a corresponding acknowledge signal, driven by the receiver.

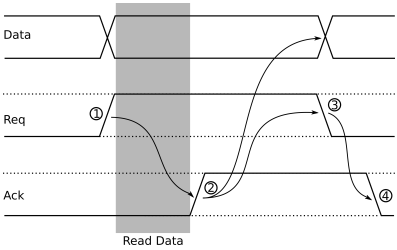


Figure 3.4:
Flow-Diagram of the employed four-phase handshake in event-based communication via parallel AER

When data is available, the transmitter sets the request *Req* to a 'high' level. As Figure 3.4 illustrates, the receiver reacts to a rising flank in the request *Req* by pulling the acknowledge *Ack* to 'high' as well, as soon as the data is latched. This, in turn, causes the *Req* to be dropped to 'low' again. The handshake is ended by lowering the *Ack* again, the signal that the receiver is ready for the next transmission [Deiss et al., 1994].

Historically, the AER protocol used for event-based communication with the processor was developed for one-to-one communication between a source and a single destination, and it is easy to see that AER forms a bottleneck when more complex networks with one-to-many mappings are implemented. In this scenario, large fan-out causes a substantial increase in traffic.

Solutions to this problem range from time-sharing neurons (TrueNorth, IBM [Merolla et al., 2014b]) to local routing and global broadcasting (CxQuad [Indiveri et al., 2015]) to only broadcasting [Bamford et al., 2010].

Besides posing a data-throughput problem, large fan-in and fan-out in the topology of the network also make it necessary

to provide a sufficient number of synapse circuits, but this restricts scalability. This fact is illustrated by the ROLLS neuromorphic processor [Qiao et al., 2015]. Here, a central event router, external to the IC, implements the topology of the network according to a routing table which stores the network's adjacency matrix A. In order to support arbitrary topologies, A, and the corresponding array of synapse circuits is laid out with the dimensions $N \times N$, where N is the number of neuron circuits present. This design choice allows to potentially connect every neuron with any other. Additionally, $S \times N$ elements are required to allow arbitrary connection of the network to an input vector S. The choice to support dense connectivity implies that the number of synapses and routing memory scales exponentially with the number of neurons. As is visible from the annotated micrograph of the ROLLS chip (see Figure 3.3), this results in a layout, where most of the ICs real-estate is used to implement synapse circuits.

At run-time, a spiking neural network's adjacency matrix typically remains structurally unchanged. In this structurally static scenario, off-chip plasticity rules only change the elements values that store the weights of connections between a minimum and a maximum value during learning. Biological assemblies of neuronal cells, however, undergo a constant modification by synaptogenesis and destabilization of synapses. This dynamic reconfiguration of the networks topology occurs not only during development but also even in adult neuronal populations. This structural re-arrangement allows the network to limit fan-in and fan-out by avoiding functional connections where they are unused.

The rationale behind the design of neuromorphic processors such as ROLLS, that store their routing table in programmable memory external to the chip, is to provide the network designer with the ability to 'wire' the neurons in a user-friendly

fashion, specific to the task at hand. This direct access, however, also opens up the possibility of reconfiguring the topology of the network during run-time, an ability that is explored throughout this thesis to explore adaptive reconfiguration of the network's topology.

Throughout this work, we made use of the *Monsterboard* PCB (see Figure 3.5), designed by *D. Sumislawaska, F. Corradi, and N. Qiao.* This solution interfaces the ROLLS neuromorphic processor with the Raggedstone Spartan6 FPGA development board [Ltd., 2010] and provides various connectors for monitoring and debugging chip performance.

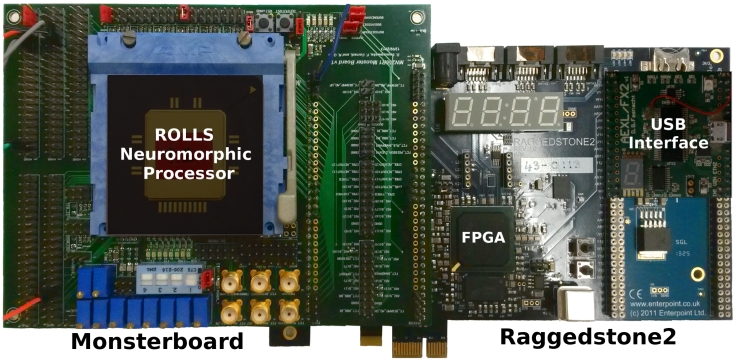


Figure 3.5:
The Monsterboard as it was used throughout this project, provides an interface between the FPGA board and the ROLLS neuromorphic processor.

The platform uses the USB interface centered around a *Cypress Fx2*, developed by [Fasnacht, 2016].

For the purpose of interfacing the neuromorphic processor with a host PC, as well as to create a platform that allows the implementation of our structural plasticity rule within a

neuromorphic system, we created a programmable System-on-Chip (pSoC), which allows online configuration of the topology of the network. The core of this system is formed by a microprocessor architecture, implemented in the FPGA used, as described below.

3.2 Programmable System on a Chip - Overview

The entire pSoC is implemented in a XILINX SPARTAN6LX45t FPGA on an Enterpoint Raggedstone 2 development board. This particular board was chosen for its number of available I/O pins, needed for the Parallel Address Event Representation (PAER) interfaces which form an interconnection with the neuromorphic processor. Additionally, it features a $1Gbit$ DDR3-Random Access Memory (RAM) used to store the topology of the network in an adjacency matrix. This data is used to route emitted source events to the specified target destination neurons.

The Xilinx Spartan 6 FPGA family is located at the lower end of the manufacturer's portfolio when it comes to resources and pricing and is aimed at applications for consumer technology and industrial automation. In the following, we show that even such moderate devices are sufficient to host lightweight pSoCs as part of neuromorphic systems. This line of devices is optimized for comparably low power consumption while featuring hardware memory controller blocks, and multiple I/O banks that support various standards like LVTTL, LVCMOS33, etc. on 296 physical pins. The particular SPARTAN6LX45t that is used in this project features 43661 logic cells, $401Kb$ of distributed- and $2088Kb$ of block RAM. The

central part of the pSoC that forms the architecture for communication and processing of events, interfacing to the neuromorphic hardware, is formed by a soft processor, the hardware description of a processor architecture that can be implemented in FPGA. In this particular case a Xilinx microBlaze Processor Intellectual Property (IP), an off-the-shelf IP core from Xilinx [XIL, 2008], was used. The choice of this particular processor was made for its extensive documentation and the well-established software toolkit for development and debugging. The soft implementation was chosen over alternative hard implementations which came to market in recent years, such as the Zynq family of devices which include Advanced RISC Machine (ARM) processors integrated beside an FPGA on a single IC [XIL, 2017] due to its flexibility and reconfigurability. The Xilinx microBlaze IP is an FPGA-optimized 32-bit Reduced Instruction Set Computer (RISC) processor that stores data and instructions in local memory (Block RAM). Nevertheless, the underlying architecture is Harvard since different address spaces are used. However, in the implementation at hand, part of the processor memory is located in the Raggedstone's DDR3 to save Block RAM resources. To aid software development, the Xilinx Microprocessor Debug Module (MDM) supports breakpoints and provides a Joint Test Action Group (JTAG) interface, through which the Xilinx System Debugger (XSDB) software can connect to the processor for code execution at the individual instruction level. As part of the Xilinx workflow, the Embedded Development Kit (EDK) interacts with the tools described and presents a single interface for pSoC design and synthesis. The Processor's IP core is provided as a parameterizable netlist and can be configured through the EDK. With the help of the software development kit provided, C code can be compiled and implemented in the bitstream that configures the FPGA. The

compiler used here is a customization of the GNU Compiler Collection (GCC). The Microprocessor Hardware Specification (MHS) instantiates the pSoC's components and specifies their interconnections and the Microprocessor Software Specification (MSS) file. Xilinx Synthesis Technology (XST) is used for the synthesis of the specified system.

The structure of the workflow enables developers without a background in VLSI to comfortably design and debug algorithms for plasticity rules and preprocessing of external event-streams.

The microBlaze IP core is highly customizable, for instance multipliers, dividers and floating point support can be enabled. The processor's instruction set architecture can optionally implement multiplications and divisions in hardware, yielding a latency of 3 clock cycles for multiplication operations and 34 clock cycles for division operations. [XIL, 2008]. In order to reduce resource utilization, these features were not made use of. At maximum, the processor supports a clock-rate of $150MHz$, However to allow compatibility with the DDR3 RAM $100MHz$ was chosen. The processors pipelined instruction management allows execution of one instruction per cycle while subsequent instructions are fetched and decoded in the same cycle. The remaining majority of resources that are still available after including the processor in the FPGA's logic can be used to implement peripheral digital circuitry.

3.3 Programmable System on Chip - Architecture

In order to process data in real-time, the microBlaze processor is used 'bare-metal' without an operating system that might

introduce time-variant delays and would be resource intensive. Approximately 1,656 of the available 6,822 slices available in the Spartan 6 (24%) are dedicated to the System-on-Chip (SoC), so that further standalone digital circuits may be run beside the SoC without interference.

On the software side, the code-repository that includes the basic functionalities for interfacing the neuromorphic processor is organized in a set of lower-level drivers for each peripheral IP core and a higher level wrapper that aims to simplify the usage of the drivers and provide functions to stimulate and set specific synapses via PAER, program biases, etc.

Communication within the pSoC is established through an AXI4 bus, an industry standard that allows easy integration of off-the-shelf peripheral IP cores as well as custom cores to expand the flexibility of the system with blocks such as spike-train generators, etc. The AXI4 parallel interface supports configurable bus widths and different modes of operation like streaming of data and master-slave communication. AXI4 was used as a processor-bus throughout the system to interconnect the processor with the peripheral blocks at a bus speed of 100MHz. Apart from the primary interface bus, a second bus interfaces to the DDR3 memory in a one-to-one connection to reduce the amount of traffic on the interface bus and avoid lower data rates. Here the bus-width used was $32bits$ to match the processor's register size.

Peripherals are designed in a modular fashion and can be added in an application-specific manner, e.g., for incorporating direct pAER sensor input. The processor can be used in both a standalone configuration, where biases and stimulation are applied automatically (e.g., for the use in autonomous robotic agents) or as a bridge which parses input from a host PC or pre-processes event-data before sending it to the host for logging and visualization.

Both use-cases were verified in several scenarios

- Interfacing of the RAMP chip for driving memristors (bridge) [George et al., 2017]
- Synapse characterization (autonomous) [George and Indiveri, 2016]
- Event-filtering application (autonomous) (see below)
- Interfacing the RAMP chip's ADC (bridge) (Appendix A.5)

The main application in the present context is the execution of algorithms involved in activity dependent structural plasticity. Here, the input event streams provided by the host PC were generated to create controllable test-scenarios. In contrast to regular SoCs, in the present setup the peripheral blocks that interface to the main processor are described in VHSIC Hardware Description Language (VHDL) and are thus re-configurable and easily extendable to meet the requirements posed by particular experimental setups. These peripherals are split into two parts, a generic Advanced eXtensible Interface (AXI)4 slave entity and the dedicated entity which gives the peripheral its specific functionality and interacts with the AXI4 slave over a FiFo interface.

The AXI4 slave entities are assigned base-addresses in the address space, and their individual registers are addressable by adding an offset to these base addresses.

Figure 3.6 details the organization of the address space used in the system. The processor can interact with the peripherals through pointers which are passed to drivers which are written in C. The corresponding address-register is accessible by the dedicated part of the peripheral.

microblaze_0's Address Map			
microblaze_0_d_bram_ctrl	C_BASEADDR	0x00000000	0x00007FFF
microblaze_0_i_bram_ctrl	C_BASEADDR	0x00000000	0x00007FFF
RS232	C_BASEADDR	0x40600000	0x4060FFFF
axi_intc_0	C_BASEADDR	0x41200000	0x4120FFFF
debug_module	C_BASEADDR	0x41400000	0x4140FFFF
axi_s6_ddrx_0	C_S0_AXI_BASEADDR	0x60000000	0x6FFFFFFF
masterip_0	C_BASEADDR	0x70A00000	0x70A0FFFF
paertoaxi_0	C_BASEADDR	0x70E00000	0x70E0FFFF
paeroutput_0	C_BASEADDR	0x74600000	0x7460FFFF
fx2toaxi_0	C_BASEADDR	0x7DE00000	0x7DE0FFFF
biasgen_0	C_BASEADDR	0x7E800000	0x7E80FFFF
Unmapped Addresses			
axi_s6_ddrx_0	C_S1_AXI_BASEADDR	0x80000000	0x8FFFFFFF

Figure 3.6:
Address map of the peripherals interacting with the MicroBlaze Processor

Figure 3.7 is taken from the Xilinx EDK, visualizing the various system components and their interconnections. The same strategy of address-mapping is employed when using the external DDR3 RAM: the system's memory range is expanded by the external RAM, and a certain address range is reserved for addressing its contents. This way the user does not need to know which specific type of memory is used unless a specific memory range is targeted using a pointer offset.

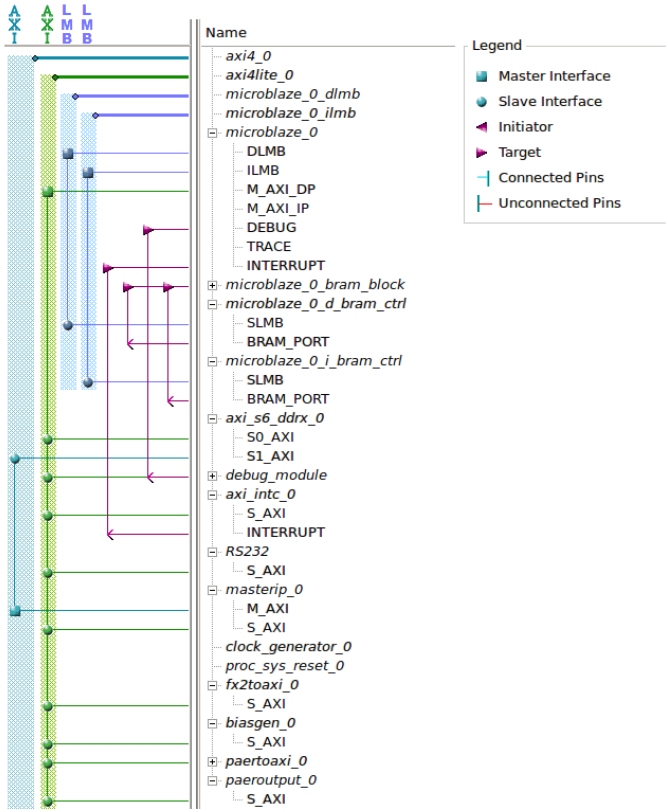


Figure 3.7:
Screenshot of the Xilinx Embedded Design Kit (EDK), detailing the organization of peripherals in the pSoC.

VHDL entities peripheral to the microprocessor can trigger interrupts of the length of one clock cycle, which trigger an interrupt service routine in the processor's code. In order to interface with the asynchronous handshake used to com-

municate between low-level VHDL entities and neuromorphic ICs, an acknowledge signal is passed down from the AXI slave to the peripheral and is there directly passed through to the state-machine that interfaces to the next entity downstream, see Figure 3.8.

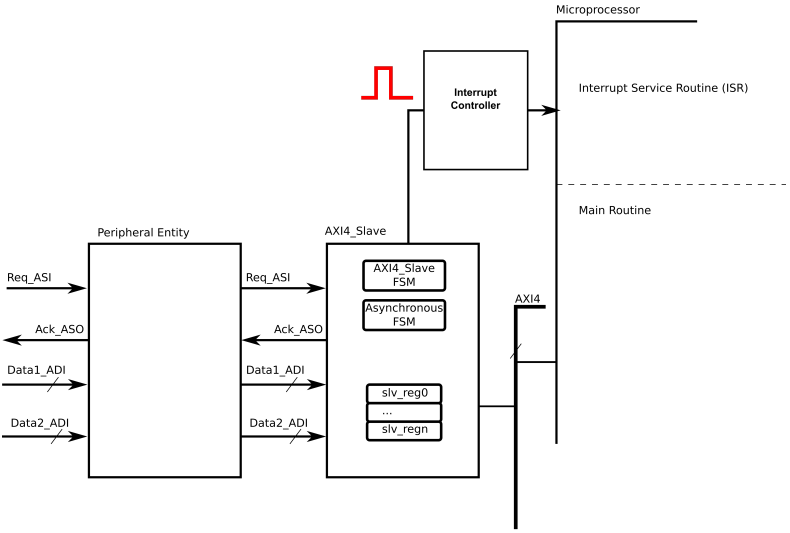


Figure 3.8: *Illustration of the implementation of the interrupt*

Interrupt service routines can be used to process data in an event-driven fashion. Alternatively, a periodic triggering through a timer can be used to leave the main body of instructions and to set the program pointer to the beginning of one of several interrupt service routines. Using interrupts allows freeing up capacities for ongoing supervision of event-streams and adaptive reconfiguration of the network topology instead of polling for incoming events. The modular design of the system features IP cores for providing an interface to neuromorphic ICs following the Parallel Address Event Rep-

resentation specification. Relative or absolute timestamps for received events can be computed in the low-level VHDL to add information which is necessary for synaptic plasticity schemes. The temporal resolution for this is configurable to a maximum of $32bit$ with an accuracy of $2ns$. The neuromorphic IC contains integrated digital-to-analog converters for generating bias voltages that serve to adjust parameters such as the refractory period or spike threshold in the on-chip neurons and synapses. To interface to this circuitry, a bridge from the AXI4 processor bus to the serial BiasGen protocol was implemented. This bridge allows the initialization of biases from the processor as well as the simulation of neuromodulatory effects by, for example, manipulating the leakage current supplied to the neuron circuits.

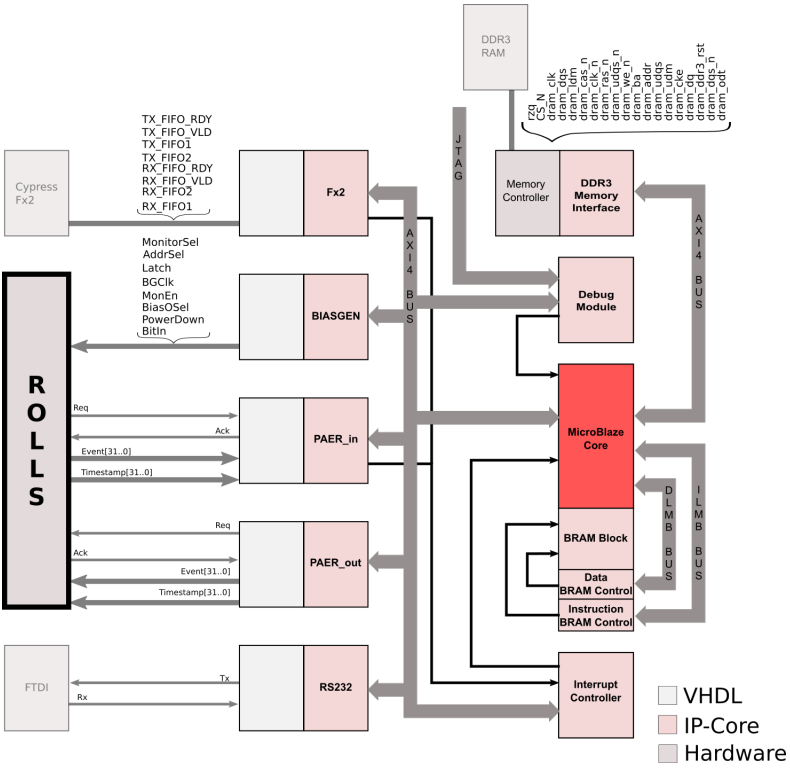


Figure 3.9:
Schematic overview of the programmable system-on-chip designed to interact with the ROLLS neuromorphic processor and a host PC

Apart from application-specific components, an interface from the processor to USB for the communication of input- and output data to the system, and a controller for the external DDR3 RAM for storage of event-routing tables are also provided (see Figure 3.9).

3.4 The Host PC Interface

The interface for logging data and externally writing configuration and stimulation data to the system is provided through a Cypress Fx2 IC that was already used in previous work. To make use of this interface within the frame of the present work, the driver was extended to provide single-event output where it previously was only possible to receive events buffered in packets of a certain length. The driver is interfaced through a set of C++ functions that can be embedded within Python. In the Python level, stimulus generation or replay of logged event-streams to the system, as well as logging and analysis functions for the output from the system are implemented and used in the verification described below and in later stages of the project.

3.5 Application Example for the Co-Processor Architecture

As an initial step in the development of the neuromorphic system, a rudimentary core system consisting of the 'soft' implementation of a processor in the backbone FPGA and a minimum number of interfaces was developed with the aim of establishing whether the pSoC approach was capable of performing the computation involved in later activity dependent structural plasticity algorithms in the limited time between incoming events, and to establish whether the necessary fundamental building blocks such as the support for multiple interrupt instances was available to the necessary degree.

To serve as a toy problem, a $128 \times 128 pixel$ DVS [Lichtsteiner et al., 2006] was used as an event source for the pSoC. The

DVS provides an address event if a positive or negative change in light-intensity is detected in a pixel, and thus performs as an asynchronous vision sensor, as distinct from a conventional frame-based video camera. The task selected was to create an on-line filtering mechanism that suppresses background noise in the sensor's event stream.

Since the same parallel AER interface is used that is also implemented in the ROLLS neuromorphic processor, the DVS here forms a substitute event source that is easy to control via the visual stimuli provided.

The goal of this evaluation experiment was to verify that the soft processor architecture and the bus architecture implemented are capable of processing incoming events at high rates, in real-time.

The detection and suppression of spontaneous firing in the DVS sensor-array was selected as a computational task. The stimulus in this case was a toy racing car game where a model car is traveling on a fixed trajectory and generates events as contrast changes are produced by the motion of the car.

To reliably suppress events that are sent from continuously active 'hot' pixels in the sensor array, a center of mass for events that occur in a time window was computed. Suppression of events is performed as a function of the distance to this center of mass from the cluster of events. Since the time window is implemented as a sliding window, the overall algorithm can still be considered event-based and all the computations involved are performed in the inter-spike interval between incoming stimuli. The Raggedstone board's DDR3 Static Random Access Memory (SRAM) serves to provide the necessary memory resources involved in the filtering operation. After the filtering stage, residual events are transmitted to a host PC via a USB-interface, together with relative timestamps which allow the reconstruction of the event-stream for

visualization. All the components that were developed for the system including interrupt controllers, drivers, event-handling state-machines, and internal bus-interfaces, were re-used in later stages of the project.

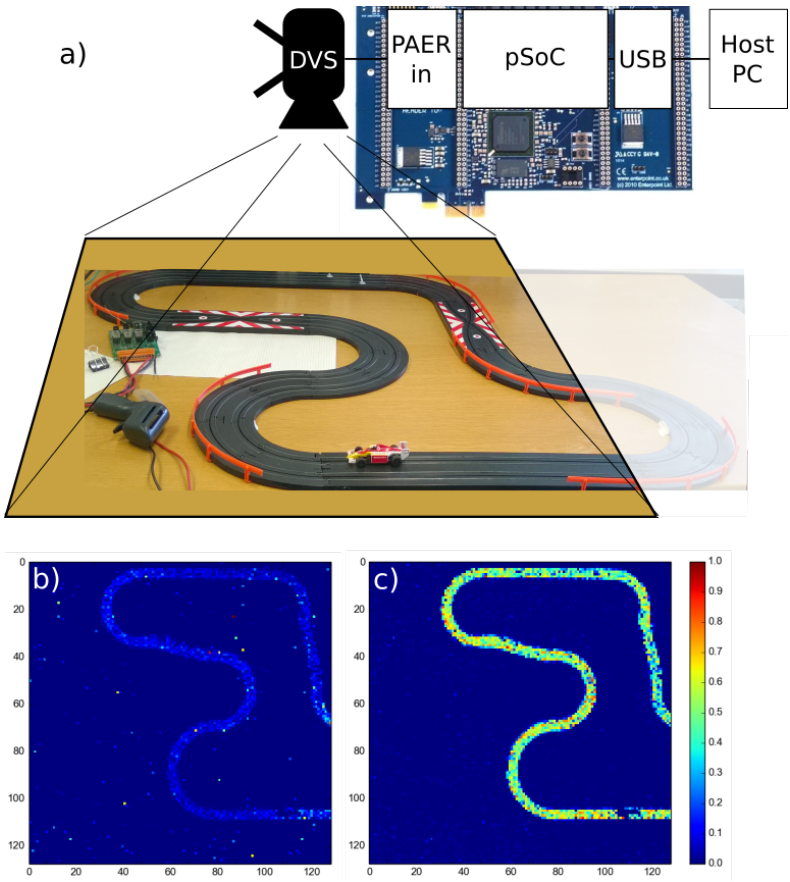


Figure 3.10:
a) Race track with DVS above, connected to the pSoC; b) Unfiltered event input as 2d histogram (normalized); c) Normalized histogram of recorded events after filtering

As Figure 3.10 shows, 'hot' pixels are successfully suppressed after the filter learned the race-track in the sensor's field of vision by clustering spatially correlated events. In the second

step, the distance of any new event to the center of the cluster was computed, and events were discarded if they were too spatially distant from the center of activity. Thirdly, a new center of activity is computed, incorporating the new event. Doing so requires the triggering of an interrupt on the request from the DVS and performing the filtering operation that updates the DDR3 memory of the system.

3.6 Summary

In summary, with the help of a toy problem, we demonstrate the pSoC architecture described above. The system provides a flexible and user-friendly solution for the implementation of event-based algorithms close to the neuromorphic hardware to which it is interfaced. Through the use of parallel buses, the bandwidth bottleneck is circumvented, and processing is performed in a way that supports standalone systems and robotic applications without a host computer. Choosing for a soft implementation of a microprocessor within the FPGA increases the number of possible use-cases because the system can be configured to an almost arbitrary extent, only limited by the FPGA's resources. This system provides the target architecture for the development of structural plasticity algorithms that dynamically adapt network topology to the stimulus statistics presented to the neuromorphic processor.

4

Modeling Activity Dependent Structural Plasticity

In our aim to create a model of structural plasticity in a neuromorphic system, the choice of a deterministic model is beneficial, since it allows the setting of hard limits on the fan-out and fan-in of the neurons involved. This makes it possible to keep the network's adjacency matrix A, which encodes the topology of the network, constantly sparse, even as the network undergoes learning. This sparsity makes A compressible to save memory resources in off-chip routing. A stochastic model that relies on imposing activity independent probabilities for synaptogenesis and synapse pruning would not allow the maintenance of a constant number of synapses at any given time. Using structural plasticity promises to lift the scalability constraints which come with the number of synapses which need to be implemented in hardware. Future chip generations may implement larger numbers of neurons N with fewer synapses than $N \times N$ where the network reconfigures itself during training to place connections where necessary.

Structural plasticity is also capable of reducing the communication bottleneck over the shared parallel AER bus. This bottleneck is a result of the necessity of connecting neuromorphic neurons to one another through event routers to create a network. Use of shared buses for event communication implies an increase in traffic as the connectivity in the network

increases with the network's size. A centralized routing with a single router that implements the entire network's topology results in a large amount of traffic where a single source event is causing the transmission of several destination events targeted at postsynaptic neurons when it is performed in a point-to-point fashion.

In previous work, attempts at imposing fan-out constraints to the neurons and use of special broadcasting-events have been made by several authors to approach this problem. The introduction of hierarchical routing as used by [Merolla et al., 2014a], [Indiveri et al., 2016] provides a solution by introducing the notion of local broadcasts within subpopulations of neurons. Destination neurons share the same destination address, and a single event can be used to stimulate a population. A more extreme case is presented in [Bamford et al., 2010] where one-to-one communication with neuron-specific destination addresses is completely replaced with broadcast events. In this work, the individual synapses are programmed in hardware to accept or ignore these events. In this way, arbitrary fan-outs can be realized. The downside of this approach is that memory has to be introduced locally at the synapse circuit, to store the source identifiers (IDs) it is to respond to.

Similar to the outlined approach, restrictions in both fan-in and fan-out can be implemented through structural plasticity rules, with the advantage of bringing an aspect of biological plausibility into neuromorphic systems at the network-level. This would allow reducing the amount of engineering that has to be performed prior to runtime in setting connections. Rather, the network would self-organize in an unsupervised fashion to accommodate the restricted number of connections in the most optimal fashion.

Another implementation that focuses on neuromorphic hardware is presented by Hussain et al. [2013]. Here, the neuro-

morphic neuron's synaptic inputs are grouped into dendritic branches that exhibit nonlinear activation. The authors implement a classifier that iteratively prunes the weakest synapses from one set of connections and replaces them with better-performing members of a second, randomly chosen 'silent' set. This replacement is performed locally on the dendritic branch. The result of this operation is quantified by the error metric of the classification and re-location is repeated if no performance increase is observed.

In other related work, Deger et al. [2016] introduce structural plasticity to explain the biological evidence for the bimodal distribution of multi-contact synapses between pairs of neurons. Spiess et al. [2016] also show the de-noising and compressing effects that structural plasticity has in large-scale networks.

From the literature review in Chapter 2, it became clear that long-term synaptic plasticity and structural plasticity are strongly interdependent due to the role of the CaMKII signaling pathway that mediates both morphological changes and changes in synaptic efficacy.

For this reason, in this project, we made the choice to base the structural re-configuration rule on a model of synaptic plasticity that captures changes in efficacy, and to extend this model to be capable of displaying activity dependent changes in spine motility and synaptogenesis. Contrasting this focus on the links between the mechanisms for structural- and synaptic plasticity, a different approach is given by Butz and van Ooyen [2013]. Here, the authors take the firing activity as a regulator of the synaptogenesis-rate. In the case that the neurons firing rate falls below a threshold, triggers synaptogenesis both on dendritic spines and axonal boutons. The presence of both bouton and spine on a connection triggers the insertion of a synapse in this model, independent of activity. This ap-

proach was not further investigated as a potential candidate for the implementation in neuromorphic hardware, due to the lack of an explanation on how such rule could be implemented in biology. Moreover, the taken approach of coupling the processes of synaptogenesis to a hebbian mechanism allows us to benefit from the functional implications of our model, discussed in Spiess et al. Specifically, the algorithm is based on a discrete implementation of spike-timing-dependent plasticity, which modifies 'virtual' weights of the connections between a population of input sources and postsynaptic neurons. These weights have no functional impact and should be interpreted as an indicator of the stability of a synapse, rather than of its efficacy. The dendritic spine dynamics that are summarized in Figure 4.1 are implemented as edge-cases of this plasticity rule.

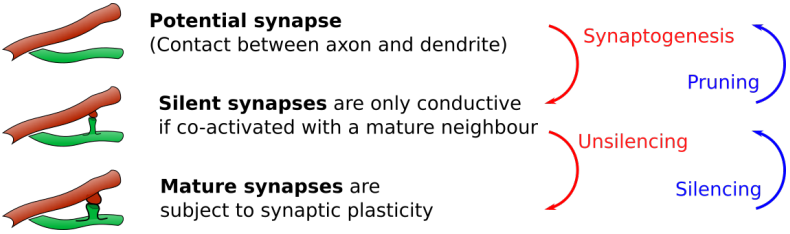


Figure 4.1:
In the following, we destinguish between potential, silent and mature synapses to capture spine synamics. The transition between these states are called synaptogenesis, pruning, silencing and unsilencing, in the following.

As the weight of a given connection undergoes modification, its value is compared to a threshold θ_{state}. As soon as synaptic plasticity depresses the connection's weight $w_{ij} + \Delta w < \theta_{state}$, its status changes to 'silent', in accordance with the observed activity dependent changes in spine morphology, described

θ_{state} : pruning threshold

if: $w_{ji} < \theta_{state}$

//start pruning counter

for t_n in $t_0, ..., t_{prune}$

if: $w_{jit_n} > \theta_{state}$

→synapse recovered

→**break**

return

if: $w_{jit_{prune}} < \theta_{state}$

→prune synapse w_{ji}

→spawn new synapse on same postsynaptic neuron

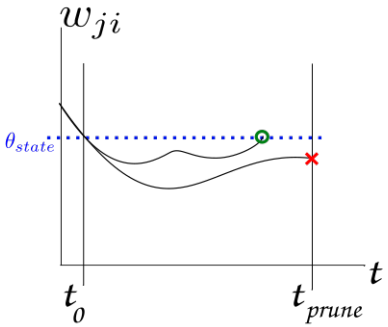


Figure 4.2:
Left: Pseudo-code for the structural plasticity algorithm; Right: Example of pruning behavior

in the previous chapter. Silent connections are in a destabilized state and get pruned after a fixed time window has passed, unless the weight recovers to $w_{ij} + \Delta w >= \theta_{state}$ (see Figure 4.2) as observed in the destabilization of biological synapses through the reduction of PSD-95. Connections that do not recover by the end of the time-window are pruned. Since there are a constant number of connections available, pruning leads to the creation of a new contact to the same postsynaptic neuron, automatically. This contact is used to send events to the previously freed up synapse circuit on chip. In this way, a constant small number of connections is maintained throughout run-time. New connections are placed randomly, following a Gaussian distribution centered around the connection with the largest weight on the postsynaptic neuron. The newly connected synapse is initialized with a small weight so that the probability of depression below θ_{state} and conse-

quent pruning is high. Newly placed synapses are therefore tested for their usability shortly after instantiation.

4.1 Applying the Computational Paradigms Identified

A translational step was necessary for the application of the identified paradigms. Where biological networks reduce the number of synapses used due to resource considerations, neuromorphic systems come with a fixed upper limit of possible connections that are achievable due to the hardwired number of circuit instances present in the processor at hand. Instead of growing new connections, the way the synapse circuits are interfaced is changed. Conceptually, this process can be described as modifying the connectivity of the presynaptic terminal to a given external or internal source, where the synapse circuit remains associated with the postsynaptic neuron. In the following, the term *connection* is used for the elements of the adjacency matrix, while *synapse* refers to the physical circuit which provides input to the neuron.

As detailed in previous sections, the environment in which the dynamic reconfiguration of connectivity is taking place consists of a processor architecture with a limited system clock frequency and limited memory resources. As outlined in Chapter 2, the majority of the allocated memory is dedicated to storing the adjacency matrix, according to which, a routing of source events to destination neurons is performed. The following introduces an algorithm that dynamically selects sources for a specified number of neurons being emulated by the ROLLS neuromorphic processor [Qiao et al., 2015].

Figure 4.2 illustrates a coarse overview of the system and

its high-level functions. These are described in detail below.

4.1.1 Initialization Prior to Run-Time

Prior to execution of the algorithm at network run-time, the weight matrix is initialized in uniformly random fashion under the user-provided sparsity constraints, so that a small number of sources $s \subset S$ of the overall available event sources is connected to destination neurons $n \subset N$. In a neuromorphic context, the allowed number of postsynaptic terminals per neuron, its fan-in d_n, is the limiting factor. d_s denotes the fan-out of a given source. In the application of the algorithm introduced for source selection, it is beneficial to introduce a limit to d_s since limiting this measure acts as a regularizer that prevents strong preference of some sources over others.

Taking both degrees into consideration, the sparsity of the corresponding topology, stored in its adjacency matrix A can be expressed as $\psi_A = \frac{d_s \cdot d_n}{|N| \cdot |S|}$. Throughout the following, $|\cdot|$ denotes the number of elements in a set, $abs(\cdot)$ denotes the absolute value of a signed variable.

To hold the user-defined constraints, an instance of the 'exact cover' problem, the outlined algorithm, a variation of Knuth's Algorithm X [Knuth, 2000], performs random weight allocation $w_{ij} \in [0, w_{init}]; i \in [0, .., \mid S \mid -1]$ and $j \in [0, .., \mid N \mid -1]$ as outlined in the following (see Algorithm 1).

Algorithm 1: Adjacency initialization

Data: $\mid S \mid; \mid N \mid; d_n;$
Result: Randomly initialized matrix $A^{d_n \times |N|}$

```
1  for n = 0 to | N |
       /* Find 64 unique random sources from
          LFSR32                                   */
2      im = 0
3      For in = 0 to (in =| S | −1) ∧ (im = d_n − 1)
4          rn =| S | −in
5          rm = d_n − im
6          if rand()%rn < rm then
7              found_pre[im++] = in + 1
8          end
9      end
       /* find 64 UNIQUE random numbers from
          LFSR32 for post                          */
10     im = 0
11     For in = 0 to (in =| S | −1) ∧ (im = d_n − 1)
12         rn =| S | −in
13         rm = d_n − im
14         if rand()%rn < rm then
15             found_post[im++] = in + 1
16         end
17     end
```

```
19
    /* remove ascending order                                    */
20  for i = 0; i < d_n; ++i do
21      randIdx = rand()%d_n;
22      t = found_pre[i];
23      found_pre[i] = found_pre[randIdx];
24      found_pre[randIdx] = t;
25      t2 = found_post[i];
26      found_post[i] = found_post[randIdx];
27      found_post[randIdx] = t2;
28  end
    /* place synapses according to generated
       pre-post pairings                                         */
29  for nn = 0; nn < d_n; nn++ do
30      randpre = found_pre[nn];
31      randpost = found_post[nn];
32      synapse.state = mature
33      synapse.pre = randpre
34      synapse.post = randpost
35      synapse.w = W_init
36      synapse.idx = nn
37      matrix W ← synapse
38      boutons_per_source[synapse.pre]++
39  end
40 end
```

In this context, *rand*() is a random number generator based on a 32 bit Linear Feedback Shift Register (LFSR) with taps at bits 32, 16, 3 and 2 that was implemented by hand to make it available on the processor platform. In future work, random number generation could be outsourced to a dedicated peripheral IP-core.

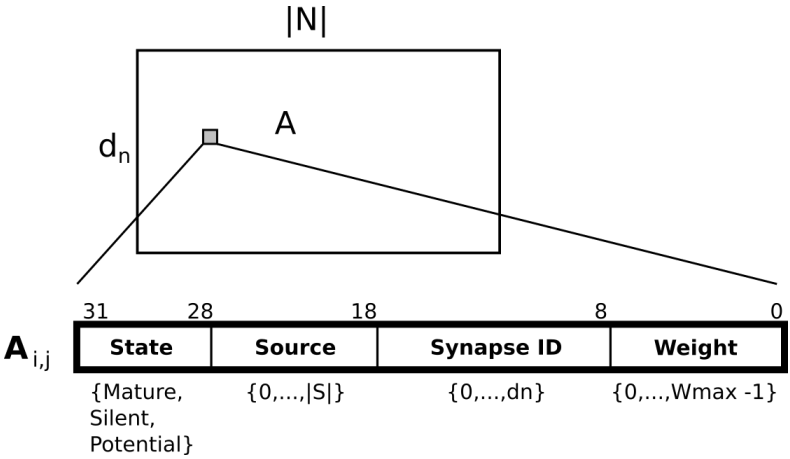


Figure 4.3:
Data structure used in the microprocessor implementation of activity dependent structural plasticity.

As Algorithm 1 illustrates, the first step is to generate d_n unique random pairs of source and destination identifiers for any given postsynaptic neuron. These first are generated in ascending order, after which they are shuffled randomly and finally included in the data structure.

To reduce time to convergence to an expected connectivity, all connections are initialized as mature with the minimum weight of 1.

Furthermore, memory for the STDP and timeout shift-registers is allocated and initialized with a 'magic' word interpreted as an *empty* entry.

The data structure used for storing the sparse adjacency matrix is an array of $32bit$ unsigned integers, where a single entry stores the state, source, target synapse ID and weight of the connection, see Figure 4.3.

The rationale behind explicitly specifying the synapse ID was that in this fashion maximally matched or mismatched synapses can be used. In future work, the methodology for synapse characterization in Appendix A.3 may be used for the choosing of synapses with different characteristics, induced by mismatch. This step would allow morphological effects to be represented by selecting differently behaving synapses for apical and distal connections.

Support functions for reading and writing to the data structure increase usability by unpacking the $32bit$ data into discrete variables within a *struct*.

In total, the use of a dense representation of the sparse connectivity allows a decrease of the memory used from $\mid S \mid \times \mid N \mid \times(\log_2\lceil W_{\max}\rceil + 4)bits$ down to $d_n\times \mid N \mid \times(\log_2\lceil W_{\max}\rceil + 4)bit$ The downside of this scheme is, however, that for routing purposes, an exhaustive search for the source information within the matrix has to be performed. So, to address this issue, a second matrix of dimensions $\mid S \mid \times d_n$ is introduced that maps source to destination. Every of the d_n elements per source stores one destination neuron address. An evaluation of the impact of this design consideration is presented in Chapter 5.

4.1.2 Implementation of Spike-Timing-Dependent Plasticity

Rules such as the one proposed by Fusi [Brader et al., 2007] allow the implementation of plastic synapses in neuromorphic hardware because of their ability to be represented by bistable circuits that store the synaptic memory. This makes more power intensive alternatives for long-term memory, such as floating gates, unnecessary. Here, the update magnitude and

polarity of the weight update depend only on the state of the post-synaptic neuron, measured at the time of arrival of a new pre-synaptic spike. A general approach for implementing this type of synaptic plasticity, using memristor crossbar arrays, was described in [Mostafa et al., 2016] and [George et al., 2017]. Let $\mathbf{s}(t)$ be a vector denoting the state variables in the post-synaptic neuron at time t. At a pre-synaptic spike at t_{pre}, the synaptic weight X is updated according to:

$$\mathrm{X} \leftarrow \mathrm{X} + r(\mathbf{s}(t_{pre})) \tag{4.1}$$

where r is a function that yields the synaptic weight increment or decrement as a function of the post-synaptic neuron's state.

The neuron's state variables that are relevant to the plasticity rule are $\mathbf{s}(t) := (V_{mem}(t), C(t))$ where $V_{mem}(t)$ is the membrane potential and $C(t)$ models the intra-cellular calcium concentration [Brader et al., 2007] described by the following equations (see [Mostafa et al., 2016] for reference and description of the implementation in hardware):

$$\frac{d\mathrm{C(t)}}{dt} = -\frac{1}{\tau_{\mathrm{C}}}\mathrm{C(t)} + J_{\mathrm{C}} \sum_i \delta(t - t_i) \tag{4.2}$$

where J_{C} and τ_{C} are the magnitude of the calcium influx during each spike and the time constant respectively, and t_i is the time of the i^{th} spike emitted by the neuron. The synaptic weight update function r (see Eq. 4.1) is given by:

$$r(\mathbf{s}(t)) = \begin{cases} a & \text{if } V_{mem}(t) > \theta_V \text{ and } \theta^l_{up} < C(t) < \theta^h_{up} \\ -b & \text{if } V_{mem}(t) \leq \theta_V \text{ and } \theta^l_{down} < C(t) < \theta^h_{down} \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

Where θ_V, θ^l_{up}, θ^h_{up}, θ^l_{down}, and θ^h_{down} are the parameters of the plasticity rule and a and b are positive constants (see [Mostafa

et al., 2015] for more details regarding the plasticity rule or [Chicca et al., 2014] [Noack et al., 2015] for CMOS implementations). According to Eq. 4.3, the neuron can be in one of three plasticity states: the potentiation state, the depression state, or the neutral state. The neuron's plasticity state determines whether afferent synaptic weights will be potentiated, depressed, or left unchanged when a pre-synaptic spike arrives.

Although the neuromorphic processor we use in this project supports this behavior, we focus on a more conventional implementation of synaptic plasticity in a digital processor architecture. This decision was made on the basis that the primary purpose here, is the driving of structural reconfigurations in the routing of sources. In fact, the digital synaptic plasticity implementation we present has no direct impact on the elicited EPSC amplitude, rather, the computed weight is 'virtual', and its purpose is to determine which synapse to connect to which source of events.

The algorithm which updates the weight of a connection in a spike-timing dependent way is given in this section (see Algorithm 2). The synaptic weight here is a cumulative measure that combines quantal size, vesicle release probability, number of AMPA receptors in the Postsynaptic Density (PSD) and their conductivity to represent the 'virtual' synapses efficacy. Weight updates are triggered both by periodic timer-events and by incoming address events. In the triggering of a timer-event, a 'magic' value that represents the absence of an event at the current time step is given as an argument to the algorithm. The triggering of the PAER Interrupt Service Routine (ISR) causes the event passed to be flagged as 'post-synaptic' by setting a bit outside of the address range of the neuron population. This allows later disambiguation of pre- and post events within the plasticity rule. Presynaptic events from other interfaces, e.g., the USB-peripheral are directly for-

warded to the algorithm. The weight update rule is closely related to the commonly used STDP formulation by Song and others [Abbott et al., 2000]. The version implemented is based on a Shift Register (Shift Register (SR)) that performs a windowed function on the event-stream. As a new address event or timer event enters the SR on the first position, all other elements are advanced by one position.

As Figure 4.4 illustrates, whenever a new event enters the SR, all entries are analyzed with respect to the center element. The notion of time is encoded in the position of the elements in the SR. If the neuron ID which is currently in the center position of the SR is flagged as a postsynaptic event, the adjacency matrix is used to search for presynaptic partners that are present in the SR. The weight update on a synaptic connection which is mature depends on the position of the presynaptic activation in the SR relative to the center-event in the center position of the SR. Depending on whether the presynaptic activation happened before or after the postsynaptic event, LTD or LTP is applied. Figure 4.4 illustrates the SR implementation of synaptic plasticity that causes LTP and LTD.

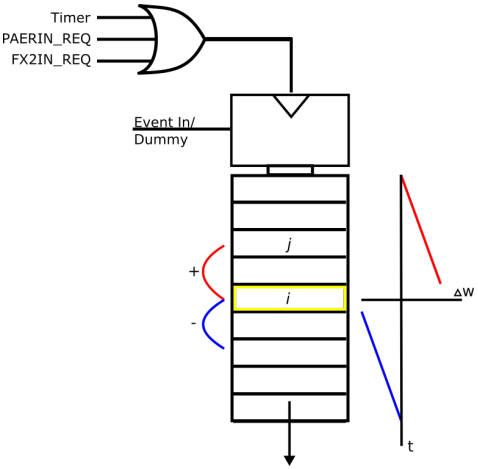


Figure 4.4:
Shift register implementation of synaptic plasticity. The notion of time is implemented through the shift-operation, driven by requests from the event source or by a timer, that advance the SR periodically. In this illustration, the events i and j entered the SR on its first position and get shifted one position further as either timer-events or address-events advance the SR. Given that a mature synapse with a weight w_{ij} exists between i and j, a weight update is performed as soon as i reached the center position of the SR. The distance of the events in the SR determines the weight update, by determining the address in the plasticity lookup-table that stores the appropriate Δw. As the SR is advanced further, and j takes the center position of the SR, a depression of the weight w_{ji} is performed. However, given the imposed sparseness of the networks adjacency, it is unlikely to find a situation where both synapses are in a mature state

Since the temporal difference in spikes $\Delta t = t_{post} - t_{pre}$ is heavily discretized by this implementation, and furthermore, the weight resolution is limited to $8bit$ due to memory constraints, a look-up table serves as a reference to determine the weight update.

Algorithm 2: Synaptic plasticity

Data: *evt* : { source event; pre-/post-synaptic or timer event}
Result: *void*

```
/* Update the plasticity SR                                    */
plasticity_sr_center = len_plasticity_sr / 2
for i = 1 to len_plasticity_sr − 1
    plasticity_sr[i] ← plasticity_sr[i − 1]
plasticity_sr[0] ← evt
/* Compute weight update                                       */
if plasticity_sr[plasticity_sr_center] is postsynaptic
    for l = 0 to len_plasticity_sr − 1
        if plasticity_sr[l] is presynaptic
            /* Read synapse struct from weight
               matrix, see Figure 4.3 for
               reference                                       */
            synapse = find_synapse(plasticity_sr[l], ...
            plasticity_sr[plasticity_sr_center])
            if synapse.state = 'mature'
                w_new_temp =
                  stdp_weights[l] + synapse.w
                if l < plasticity_sr_center then
                    if w_new_temp > 0 then
                        w_new = synapse.w
                        WeightHomeostasis(·, Δw =
                          stdp_weights[l], selector = 0)
                        timeout_sr[0] ← empty
                    else
                        w_new = 0
                        silence_mature_synapse(·)
                    end
```

```
21
22
23
24
25                  else if l > plasticity_sr_center then
26                      if w_new_temp >= maxW then
27                          w_new = maxW
28                          WeightHomeostasis(synapse, Δw =
                             maxW −
                             (int)synapse.w, selector = 1)
29                          timeout_sr[0]← empty
30                      else
31                          w_new = w_new_temp
32                          WeightHomeostasis(synapse, Δw =
                             stdp_weights[l], selector = 1)
33                          timeout_sr[0]← empty
34                      end
35                  end
36                  synapse.w = w_new
37                  if w_new > last_Wmax[synapse.pre]
                     then
38                      max_pre[plasticity_sr[plasticity_sr_center]] =
                         synapse.pre
39                      last_Wmax[plasticity_sr[plasticity_sr_center]] =
                         w_new
40                  end
41                  matrix W ← synapse
42              end
43          end
44      end
45  end
```

4.1.3 Boundary Cases

With the synaptic plasticity rule outlined above (see Algorithm 2) as the driving mechanisms for morphological changes in the dendritic spine, upper and lower limits to the connection weight form hard boundaries. The lower bound prevents the case $W_{i,j} + \Delta W <= 0$ and implements the threshold below which the synapse is destabilized. Synapses with a virtual weight of zero change their status to 'silent' and enter a second shift register *timeout-window SR* where they are no longer subject to synaptic plasticity but may be co-activated with mature neighbors. The upper bound $W_{\max}$ prevents memory overflow. The conditional $If\ w_{new} > last\_Wmax[synapse.pre]$ in algorithm 2 provides a resource efficient way of detecting the position of the maximum weight on the postsynaptic neuron. This position determines where to perform synaptogenesis (see Section 4.1.6).

4.1.4 Timeout Window Implementation

A second shift register is used to implement the evaluation period after which a silent synapse is pruned, and a new one is instantiated instead (see Algorithm 3). The information entered into each register is the neuron address (the column in W) and the synapse index (the row in W).

Algorithm 3: Timeout window update procedure

```
   Data: neuron, synapse
   Result: void
   /* Update the plasticity SR                                    */
 1 last = timeout_sr[timeout_window_len − 1] for
     i = 1 to timeout_window_len − 1 do
 2 |  timeout_sr[i] ← timeout_sr[i − 1]
 3 if neuron ≠ empty then
 4 |  new ← (neuron, synapse.idx)
 5 else
 6 |  new = empty
 7 if i = 0 then
 8 |  timeout_sr[0] = new
 9 else
10 |  timeout_sr[0] = empty
11 i + +
12 if last ≠ empty then
13 |  last_synapse =
   |    read_post_neuron(last.idx, last.neuron)
14 |  switch_silent_synapse(last.neuron, last_synapse)
```

4.1.5 Synaptogenesis

In the neuromorphic environment, with its fixed number of synapses, our implementation of the pruning of a silent synapse causes the direct instantiation of a new synapse. Synaptogenesis is implemented by choosing a new presynaptic partner and reusing the synapse circuit made available by the pruning operation (see Algorithm 4).

Here, a potential target location is chosen on the same postsynaptic neuron that a synapse was pruned from. This choice

is performed by locating the maximum weight and taking the center of a normal distribution from which candidates are drawn such that the probability of locating a new synapse close to a center of activity is larger than that of the use of a more 'remote' location. The notion of space here is arguably vague since we make use of point neurons that do not implement multiple compartments in their neurites or even the cable equations that would model the propagation of potentials within the intracellular medium. Nevertheless, given that the presented sources are structured, and there is a function describing these sources activity as dependent on their identity (or location to maintain the analogy to physical space), the instantiation strategy described leads to faster convergence to an ideal source selection than a purely random approach. In this process, centers of activity, expressed in large weights, are used as an indicator of where to efficiently place new synapses. As with other parts of the overall algorithm, this allocation could be performed by drawing from globally available information, however, to not loose biological plausibility, the imposed constraint in the design of the routine was the operation on locally available information on the postsynaptic neuron.

Algorithm 4: Switch silent synapse procedure

```
Data: neuron, synapse
Result: void
repeat
    if max_pre[neuron] ≠ empty then
        new_pre = rand(normal, μ = max_pre[neuron])
    else
        new_pre = rand(normal, μ = 512)
    if boutons_per_source[new_pre] ≥ d_s then
        break
    else
        for i = 0 to d_n − 1 do
            if W[i][neuron].pre = new_pre then
                break
until new_pre not in use;
boutons_per_source[synapse.pre] − −
boutons_per_source[new_pre] + +
new_synapse = synapse
new_synapse.pre = new_pre
new_synapse.w = 1
new_synapse.state = mature
matrix W ← (neuron, new_synapse)
```

In the generation of normal distributions we exploit the central limit theorem by averaging over a number of uniformly distributed random numbers from an linear-feedback shift register (LFSR). The uniform source distribution shows a standard deviation of $\sigma_{uniform} = \frac{b-a}{\sqrt{12}}$. By averaging over $N = 4$ samples from this, we acquire samples that show a standard

deviation of $\sigma_{normal} = \frac{\sigma_{uniform}}{\sqrt{N}}$. Generating four uniform samples in the range $a = 0$ to $b = 1023$, the resulting standard deviation is $\sigma_{normal} \approx 147$. In order to regulate the winner-take-all like behavior of the algorithm and to prevent synapses from being exclusively assigned to sources with maximal connection weight, the upper bound d_s regulates the fan-out of every source. This bound is enforced through the routine that instantiates new synapses. This routine is executed whenever a silent synapse is kept within the destabilization time-window SR until it reaches the last position, without being recovered by co-activation with a mature connection to the same post-synaptic partner.

4.1.6 Silencing and Unsilencing Synapses

The only mechanism capable of advancing the maturation of a silent synapse is its co-activation with a mature synapse. To implement this, on every execution of the synaptic plasticity routine, a search for co-activated silent synapses is performed and the results are removed from the timeout SR. This process is illustrated in figure 4.5

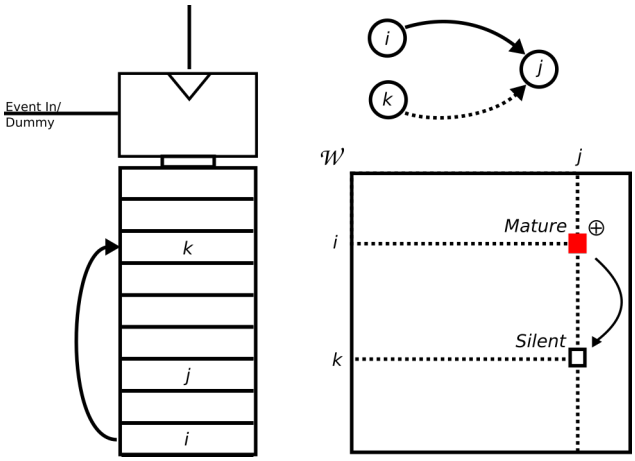


Figure 4.5:
Co-activation mechanism, implemented in the timeout shift register. Given a topology where a mature connection from neuron i to neuron j has a silent neighbor (k to j) (top right corner), an activation of neuron j causes the unsilencing of w_{kj}

The co-activation of silent neighbors (those that contain only NMDA receptors and are thus not subject to the regular STDP) is triggered by comparing a current postsynaptic event to all entries of the timeout SR. In the case that there is an entry signifying that the same postsynaptic neuron contains silent synapses, these synapses are recovered. Using this

strategy, synaptic co-activation of silent synapses (i.e., those that contain only NMDA receptors and are thus not subject to the regular STDP) with their mature counterparts is enabled. The pseudocode equivalent of this procedure is given in 5.

Algorithm 5: Synaptic co-activation of silent synapses

```
   Data: neuron, synapse
   Result: void
1  if event is post synaptic then
2      for i = 0 to |timeout_window| − 1 do
3          timeout_post ← timeout_window_i
4          if timeout_post = evt ∧ evt ≠ empty then
5              synapse_idx ← timeout_window_i
6              timeout_pre ← W[synapse_idx][timeout_post]
7              unsilence_silent_synapse
```

Unsilencing the synapse implies a change in synaptic weight, which is compensated for by calling the weight homeostasis routine. The synapse's state is also changed to 'mature' (see Algorithm 6).

Algorithm 6: Synapse unsilencing procedure

```
   Data: pre, neuron, timeout_window_idx
   Result: void
 1 for idx = 0 to d_n − 1 do
 2     synapse = read_post_neuron(idx, neuron)
 3     if (synapse.pre = pre) ∧ (synapse.state = silent)
         then
 4         target_synapse = synapse
 5     if synapse.state = mature then
 6         mature_count + +
 7 if mature_count < d_n − 1 then
 8     target_synapse.state = mature
 9     target_synapse.w = w_init
10     matrix W ← synapse
11     timeout_window[timeout_window_idx] = empty
12     WeightHomeostasis(·, selector = 1)
13     silent_tot − −
```

Silencing a synapse (*mature* $\rightarrow$ *silent*) calls weight homeostasis for compensation, however, in the case of silencing the synapse, the row information is passed to the routine within the *synapse* struct and there is no need to perform a search (see Algorithm 7).

Algorithm 7: Synapse silencing procedure

Data: *neuron*, *synapse*
Result: *void*

```
1 synapse.state = silent
2 synapse.w = 0
3 WeightHomeostasis(·, selector = 0)
4 timeout_sr[0]← (neuron, synapse)
5 matrix W ← synapse
6 silent_tot + +
```

The advantage of the described procedure over straightforward subtractive normalization by computing a value that is subtracted or added to all the weights of the postsynaptic neuron $w_{0,..,d_n-1|j} = w_{0,..,d_n-1|j} \pm \frac{\Delta w_{i,j}}{d_n}$, is that following the process of depression or potentiation of maxima or minima, low resolution single precision weights can be used to maintain a constant sum of weights on the postsynaptic neuron.

4.1.7 Homeostatic Weight Normalization

To implement the postsynaptic competition over resources identified in the previous chapter, a separate function implements a form of homeostasis in that the total sum of all weights on a given neuron is kept fixed. This function was designed to both serve the purpose of increasing synapse turnover and to implement a winner-take-all strategy that prefers those sources which receive maximal potentiation over those that remain inactive or contradictory to the postsynaptic activation. With different aims, others have implemented such a mechanism by a normalization of weights. Instead, the outlined algorithm depresses all minimum weights $\mid \Delta W \mid$ of the dendrite by a fixed amount $\Delta W_{homeo} = 1$ whenever a potentiation ΔW takes place. Likewise, in the case of depression by ΔW, the

$\mid \Delta W \mid$ maximum weights are potentiated by $\delta W_{homeo} = 1$ (see Figure 4.6).

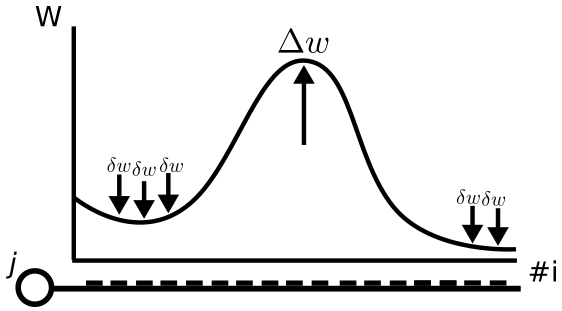


Figure 4.6:
Weight homeostasis, implemented as a subtractive normalization on the postsynaptic neuron j. For example, the potentiation of a synapse by $\Delta w = 5$ causes a depression of the $\mid \Delta W \mid = 5$ smallest weights by $\delta w = 1$.

The weight homeostasis function takes a flag as an argument that decides whether a depression of the minima or potentiation of the maxima is to be performed. According to this flag, a search for the specified number of minima or maxima is made. Following the selection of targets, similarly to the synaptic plasticity function, boundary cases are considered in the weight updates.

As outlined in the pseudo-code depiction of the homeostasis mechanism (see Algorithm 10), an argument *selector* is used to pass the decision whether to potentiate minima or depress maxima on the postsynaptic neuron. The search for the extrema in the weight distribution on the postsynaptic neuron is performed by an incremental comparison with a threshold θ. For the maxima search, θ is initialized to $last\_Wmax[neuron]$. This variable stores the index of the neuron's largest weight and is determined in the synaptic plasticity algorithm.

In the minima case, the initialization is hard-coded to W_init, the minimal value a mature synapse can assume.

In the second part of the procedure, the ΔW passed is the sum of the updates performed on neighbours of the minima or maxima, given these have the state 'mature'.

4.2 Modeling In Vitro Network Development

A component of structural plasticity less directly affected by activity, is found in the developmental processes that guide axonal growth and the branching of dendrites. *In vivo*, growth factor gradients play an important role in these processes, however, in dissociated cell cultures, centrifugation destroys these gradients, and the creation of potential synapse locations (overlaps of neurites) is primarily guided by cell adhesion molecules in the membrane of the neuron.

In the following, we present a model that provides information on the location of potential synapses, based on observations of cell morphology dynamics over the course of days of development. The aim here is to generate simulated networks that are random and non-organotypic, yet to capture experimentally observed network statistics and allow insight on the density of potential synapses over time. This model is later used in order to simulate data recorded from an MEA, to serve as an event-based input ti the neuromorphic system. Our approach does not consider potential mechanisms. Instead, we based the simulation on experimental data provided by Maschietto et al. (personal communication). The basis of this model were studies of dissociated embryonic rat hippocampal cells cultivated in vitro. These cells were grown on

the surface of a MEA. We created a second model that aims at exhibiting changes in network activity, that arise from the growth dynamics of axons and dendrites over the course of days. The model, outlined in the following, is evaluated outside the neuromorphic system on a host PC, with the goal of providing a biologically plausible stream of events which statistically changes over days as the simulated neurons radii expand and the number of potential synapses changes.

In this context, the neuromorphic system is capable of using structural plasticity for the selection of a subset of sources from this generated event stream.

In modeling purely activity independent processes, no underlying network activity influences the connectivity. Instead, the model tries to capture experimental data following data acquisition and statistical analysis of a hippocampal cell culture from mice, cultivated on the surface of an MEA. The data acquisition conducted by Maschietto et al. allowed the description of the growth of neural processes in culture in a model on PC. By using a coating with an attachment factor to ensure cell adhesion, it is assured that the involved growth processes occur in a planar fashion.

Instead of the physiological mechanisms of, e.g., axon guidance along guidepost cells and following gradients of factors secreted by the environment of the cell, only proteins in the target cell membrane inform the developing neurite whether a connection is to be established. This process was approximated by making the formation of an active synaptic connection a stochastic process. This probability derives mainly from the chance of two neurons bringing their axons and dendrites spatially close to one another. In the following, the close overlap of neurites of any two synapses is called a 'potential synapse'.

In statistical analyses of the average distance of the so-

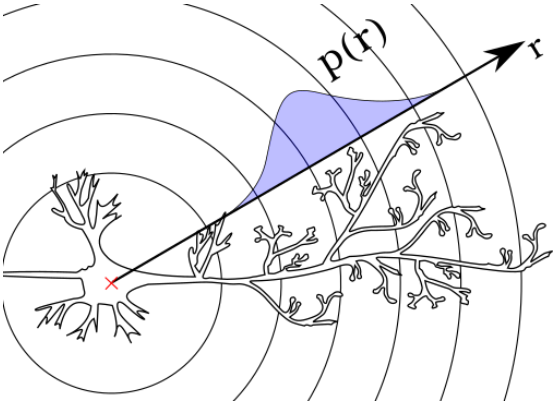


Figure 4.7:
Probability density for potential synapses as a function of the distance to the soma r, derived from Sholl analysis.

mata to each other, conducted by Maschietto et al.(See Appendix A.7), the cell-bodies are uniformly distributed in space, presumably since the cells are fully differentiated and do not undergo any migration when they are applied to the MEA surface. The initial step in the model-simulation is the placement of somata following the measured density per unit area to replicate this distribution and generate the spatial coordinates of the soma and their distances to each other. A Sholl analysis was also conducted. This method, usually used in neurophysiological studies, characterizes the neuronal morphology by placing the center of a circle on the center at the cell's soma. Now the circle's radius is expanded in regular intervals, and the number of intersections of the circle with the cell's axons and dendrites is counted.

An analysis, as described above, was conducted for two different days in vitro (DIV) by Maschietto et al., to capture developmental effects on the neuron's morphology in the Sholl

analysis. For DIV= 6 and DIV= 14, the analyses provided were approximated with a polynomial of degree $n = 6$. In order to make predictions of how this family of curves evolves for other DIV, a linear interpolation of the coefficients of the polynomial was performed. This allows the generation of curves for arbitrary DIV, following the function $p_{sholl}(DIV, r)$.

From a second experiment, that labeled synapsin, we assume that synapse density per area of cell membrane is constant. Under this assumption, it is expected that the synapse density increases as a function of the distance to the soma, as the neuron branches (See Appendix A.7)

Our estimate of the neuron's ramification over time made it possible to estimate the average density of synapses at a certain distance to the soma and at a certain point in time. For this coarse estimate, changes in the diameter of the axons and dendrites of the neurons were neglected. Given our assumptions, we can use $p_{sholl}(DIV, r)$ as a predictor for the density of potential synapses.

This approximation of intracellular volume dependent on distance to the soma and DIV was translated into a rotationally symmetric function over the two spatial dimensions of the culture to form a probability density function of potential synapses with reference to the soma. The assumption in the approach described is that the interneuronal variability in the performed analysis is small enough to assume the involved neurons are morphologically similar enough, that it is possible to compute a mean morphology by averaging over the number of intersections per radius among neurons. In this way, a two-dimensional probability density function for potential synapses was created that is dependent on time and distance to the soma. Figure 4.8 shows a cross-section of this family of functions.

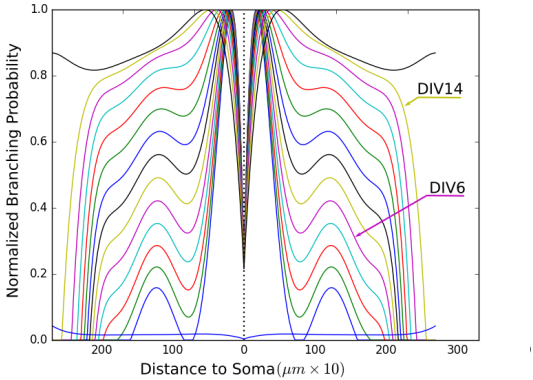


Figure 4.8:
Prediction of ramifications from Sholl analyses in in vitro cell cultures: Cross section of the kernel, abstracted from the biological data. The set of curves develops from DIV 1 to 14.

To estimate the probability of a connection between two neurons, the map of previously placed somata is convoluted with the two-dimensional probability density function to create a landscape.

In the simulation, this landscape is only partially created between any two neurons and only consists of their respective probability-density functions. To speed up the process, somata must fulfill the requirement to be closer than a specified Euclidean distance to be considered as potentially connected. The maximum in the landscape along this distance determines the probability of the connection as Figure 4.9 illustrates.

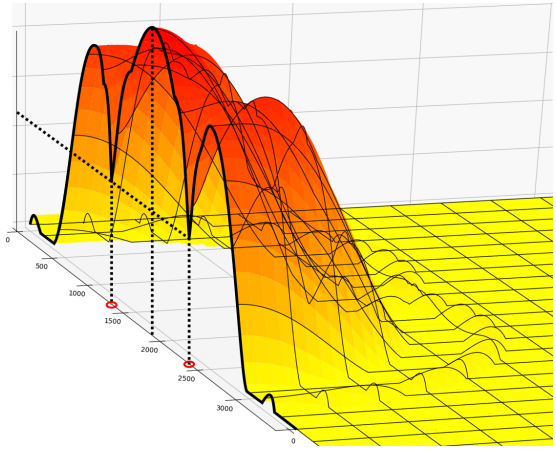


Figure 4.9:
Illustration of two kernels forming a connection probability landscape. The cross-section forms the shortest path between the two somata (red circles). The maximum in the profile along this line enters the adjacency matrix as a connection probability

The process is repeated for all possible n^2 combinations to complete the generation of a random connectivity.

An optical labeling study was conducted, that marks synapsin in the culture and resulted in samples of synapsin density at certain DIV. As a presynaptic protein responsible for the binding of vesicles in the presynaptic terminal, synapsin reliably labels active synapses and was used here to provide a measure of the average number of active synapses s that are present in the generated network. The $s(DIV, r)$ highest connection probabilities of the adjacency matrix created are selected as active synapses, where $s(DIV, r)$ is an interpolation of the measured amount of synapsin puncta for the DIV under consideration and at distance r from the soma.

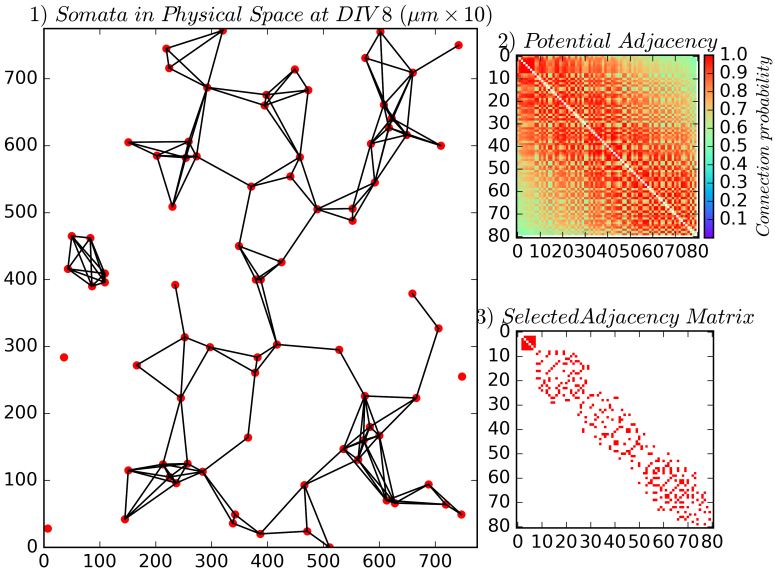


Figure 4.10:
Left: Network topology in physical space (resolution: 10μm); Top right: Adjacency matrix of potential connections

To generate spontaneous activity from the generated network, the adjacency matrix was used to create a network of Hodgkin-Huxley model neurons. This implies that electrochemical processes outside the soma are not captured. The model parameters used here were taken directly from the *BRIAN* codebase (see *http://BRIANsimulator.org/demo/*).

$$Cm = 1 * \mu F * cm^2$$
$$gl = 5 * 10^{-5} * S * cm^2$$
$$El = -60 * mV$$
$$EK = -90 * mV$$
$$ENa = 50 * mV$$
$$g\_na = 100 * mS * cm^2$$
$$g\_kd = 30 * mS * cm^2$$
$$VT = -63 * mV$$
$$aue = 5 * ms$$
$$taui = 10 * ms$$
$$Ee = 0 * mV$$
$$Ei = -80 * mV$$
$$we = 6 * nS(Excitatory\ synaptic\ weight)$$
$$wi = 67 * nS(Inhibitory\ synaptic\ weight)$$

The models equations were implemented as to match the original model proposed by [Hodgkin and Huxley, 1952].

Algorithm 8: Model equations that simulate spontaneous activity for the simulation of the MEA, implemented in *BRIAN*

1 $dv/dt = (gl * (El - v) + ge * (Ee - v) + gi * (Ei - v) - g\_na * (m * m * m) * h * (v - ENa) - g\_kd * (n * n * n * n) * (v - EK))/Cm : volt$
2 $dm/dt = alpham * (1 - m) - betam * m : 1$
3 $dn/dt = alphan * (1 - n) - betan * n : 1$
4 $dh/dt = alphah * (1 - h) - betah * h : 1$
5 $dge/dt = -ge * (1./taue) : siemens$
6 $dgi/dt = -gi * (1./taui) : siemens$
7 $alpham = 0.32 * (mV ** -1) * (13 * mV - v + VT)/(exp((13 * mV - v + VT)/(4 * mV)) - 1.)/ms : Hz$
8 $betam = 0.28 * (mV ** -1) * (v - VT - 40 * mV)/(exp((v - VT - 40 * mV)/(5 * mV)) - 1)/ms : Hz$
9 $alphah = 0.128 * exp((17 * mV - v + VT)/(18 * mV))/ms : Hz$
10 $betah = 4./(1 + exp((40 * mV - v + VT)/(5 * mV)))/ms : Hz$
11 $alphan = 0.032 * (mV ** -1) * (15 * mV - v + VT)/(exp((15 * mV - v + VT)/(5 * mV)) - 1.)/ms : Hz$
12 $betan = .5 * exp((10 * mV - v + VT)/(40 * mV))/ms : Hz$

Randomly, 20% of the synapses were initialized as inhibitory synapses to keep the network stable. At initialization, membrane voltages are set with a normal distribution $norm(\mu = 0; \sigma = 1)$ as $El + (norm() * 5) - 5 * mV$. In the following chapter, this simulation will be used in order to form a controllable input stimulus to the structurally plastic neuromorphic system in hardware, to illustrate a possible application and point out directions for future work.

4.3 Summary

To summarize, the previously identified computational primitives behind biological mechanisms of structural plasticity were applied in an algorithm, designed to operate on a digital co-processor. We also explored developmental structural plasticity observed in dissociated cell cultures and modeled network development to gain insights into how the network changes over DIVs. The routines described form the most complex variant we implemented, of the working principle outlined at the beginning of this chapter. In the following chapter, this model allows us to gain insight into how event streams from an MEA change over time. Here, we also explore the implications of structural plasticity on conventional computers and its benefits in the simulation of spiking neural networks.

5

System on Chip Implementation of Structural Plasticity

In this chapter, we report different implementations of the algorithm previously described. The order of description follows the development process chronologically. We begin with the first abstract proof of concept for the structural plasticity algorithm in a weakly typed language (*Python 2.7* and *BRIAN* [Goodman, 2009], see Appendix A.1), which served as the basis for the work by Spiess et al. [2016] and proceed to the final implementation within the pSoC previously described. We close this chapter with an evaluation of the developed approaches in neuromorphic hardware, in which controlled stimuli from a simulated MEA are applied to the neuromorphic system. Along with the steps of development, we point out the features and benefits of the application of the algorithm, also for conventional hardware.

5.1 Prototyping and Optimization in C on a PC

For the implementation of the algorithm described in Chapter 4 on the hardware platform we created, we first imple-

mented it in *C* to be simulated in conventional hardware. The goal of this intermediate step, before the final implementation in the created hardware platform, was to optimize the code in an environment that provides tools for code profiling. The GCC toolchain brings a series of analysis and debugging tools with it which are used here for optimization of the algorithm. The later step of porting the algorithm to the target platform was minimal as the used microBlaze processor is also programmable in the *C* language. At this stage, it was necessary to consider data types and structures that led to the memory saving representation of the network's adjacency matrix described earlier (see Chapter 3). Throughout the simulation in conventional hardware, the test scenario used for verification was chosen to resemble the target system as closely as possible to ease the code migration to the processor of the target system. For this reason, no external libraries were used in the core of the program. Library functions for such things as random-number generation were substituted by our own functions which can be migrated to the microBlaze processor. Since the neuron models themselves are implemented in analog VLSI in the target system, this initial *C* implementation avoided the simulation of an actual network of spiking neurons and only operated on artificially generated pre- and postsynaptic address events. Here the timing of the events from the two involved layers was set up in a fashion that supports potentiation on certain parts of the weight matrix. As the pre- and postsynaptic IDs were derived from the coordinates of pixels in a monochrome image, shown in Figure 5.1 (A), LTP should form a weight matrix that is a copy of this image. Figure 5.1 (B) shows the networks adjacency after simulation. This step makes it easy to compare the ideal weight matrix to the actual one by eye after a given number of stimuli have been presented.

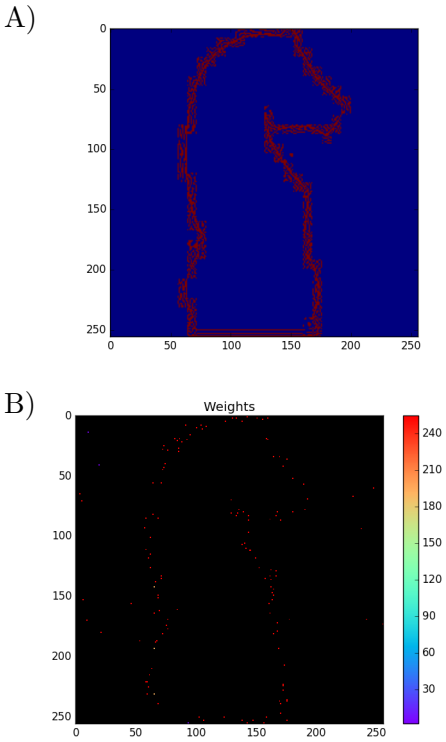


Figure 5.1:
Comparison of the Ideal Weight matrix (A) and the resulting matrix after random initialization and 5000000 events (B)

The following visualization of the algorithms call-graph 5.2 illustrates the percentage of time spent in the implemented subroutines and which of those are the most resource intensive. In an iterative process profiling tools such as *kcachegrind* [Weidendorfer, 2003] and *gprof* [Graham et al., 1982] were used for optimization:

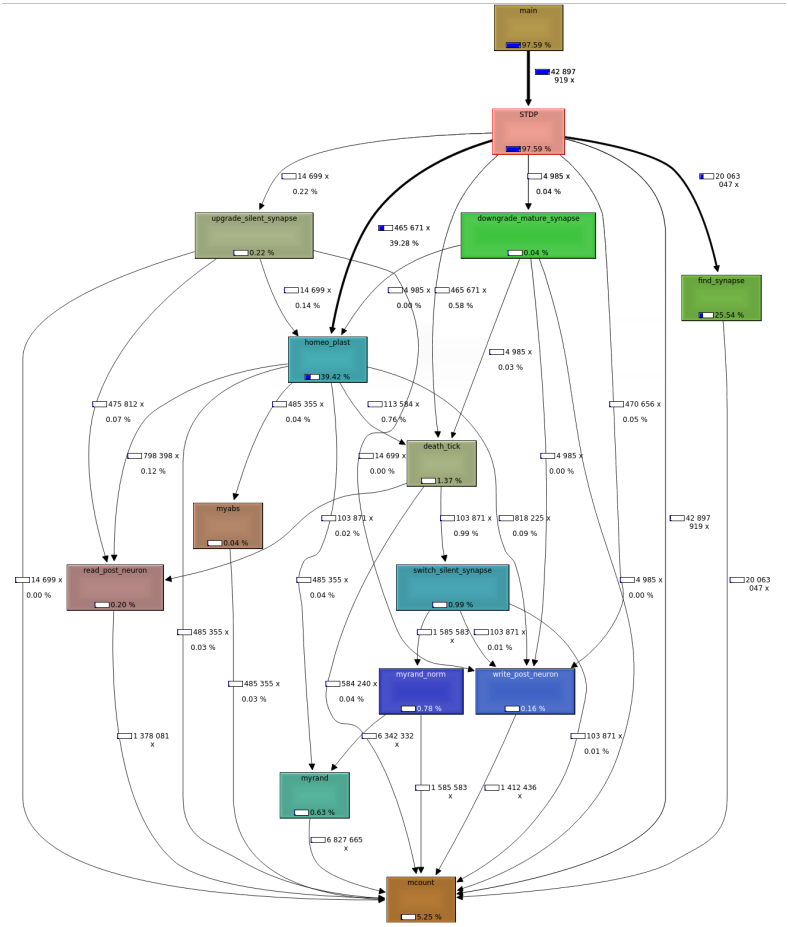


Figure 5.2:
The analysis of resource usage performed with kcachegrind illustrates the subroutines, the number of calls and the percentage of cycles spent within them

After the optimization and translation of the algorithm to *C*, we ported the code to the co-processor of the neuromorphic

system.

5.2 Microprocessor Implementation on the Target System - Verification

The implementation in the target hardware was performed gradually, first using a single neuron of the neuromorphic processor, and then expanding to include every one of the 256 available neurons.

Input to the system is provided by a vector of event sources that has a dimensionality which is far larger than the maximum degree d_n of the addressed neurons. Thus, structural plasticity can be applied in a source-selection scheme, where routing of those sources that show most correlated activity with their postsynaptic partners on the chip is performed. Since there is no internal connectivity among the chips neurons, and the synaptic weight of the synapses in hardware was set high, a repeated stimulation of a neuron by a source leads directly to the potentiation of the connection weight by causing a spiking response. In contrast, sources that show little activation lead to smaller connection weights and eventually to the pruning of the connection. Therefore, the adjacency matrix which describes the connectivity of sources to the neurons should over time form clusters of connections, where activity in the source vector is largest.

The source activity that serves as input to the system in the following experiments was generated on PC and applied via the systems USB interface. The stimulus presented consisted of 1024 Poisson-sources with a maximum firing rate of $f_{max} \approx 60Hz$.

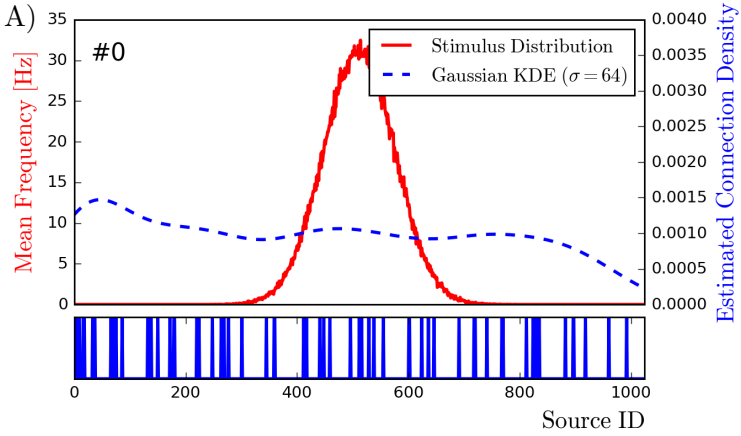
In the absence of stimuli, the 'magic' *empty* word is pro-

vided to the STDP function. Due to the nature of this experiment, it was possible to avoid reprogramming the actual hardware synapses on the chip. Instead, a re-mapping of high-dimensional sources to the targeted synapses only takes place in the networks adjacency matrix, which serves as a routing table.

As a first characterization, estimates of the time required for the code-execution were considered. An inherent result of the memory limitations on the target system, which forced the restructuring of the connectivity matrix to a dense format, is an increase of look-up time. Since a lookup now involves a search through all $d_s \times d_n$ entries in the connectivity matrix to retrieve all postsynaptic sources that require stimulation, drastic penalties in run-time are the consequence. In the dense format, the imposed delay should be more or less constant, since the memory access is more time-consuming than the stimulation itself $t_{lookup} \ll t_{stim}$. The imposed penalty can thus be approximated to be $t_{lookup} \times d_s \times d_n$, i.e., a factor d_s worse than in the case of a sparse matrix where only the row specific to the source needs to be considered in the search for connections.

5.2.1 Single Neuron Implementation in the Target System

To assess the performance of synapse allocation and to see whether connections are made at those locations where they maximally contribute to the postsynaptic neuron's firing, the single postsynaptic neuron was presented with 1024 sources. The firing rate is highest for source $\mu = S_{512}$ and is reduced following a Gaussian curve across the individual sources with a standard deviation of $\sigma = 64$. As relocation occurs, a Gaussian kernel density estimate of the synapse locations was performed as depicted in Figure 5.4.



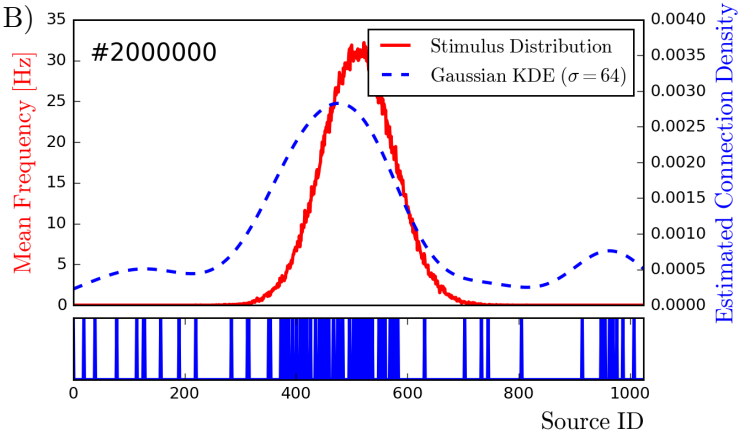


Figure 5.4:
Single neuron experiment in the neuromorphic system. 1024 sources were presented to one of the ROLLS neuromorphic processors neurons. A) Synapses are initialized uniformly. B) Change after the stimulus is presented. As an estimate of synapse density, a Gaussian kernel density estimate was performed ($\sigma = 64$) to give a qualitative measure of performance.

Comparing the synapse density estimate to the stimulus distribution, it was possible to show convergence over time. Due to the homeostasis mechanisms, a full convergence is never achieved, since there is a constant re-instantiation of synapses at random locations. The advantage of this effect is that the ongoing fluctuation of synapses implies a faster re-learning in the case of a shifted stimulus distribution. Thus the network remains constantly ready for changes and structurally plastic. Insights from this experiment are transferable to multi-neuron configurations since the algorithm was written in a fashion in which it only requires information local to the postsynaptic neuron.

5.2.2 Expansion to 256 Neurons

We tested the system in a scenario in which neuronal activity of an input population, consisting of ($S = 1024$) Poisson neurons is generated on a host PC. These sources show firing rates which are distributed in a bimodal fashion, so that two sources show a maximum rate and the rate of their neighbours decreases, following a normal distribution, with a standard deviation of $\sigma_1 = \sigma_2 = 96$ (see Fiure 5.5). In this configuration it is impossible to sample from all sources in a one-to-one fashion since only $N = 256$ postsynaptic ROLLS neurons are present and are only allowed to connect to $d_n = 32$ presynaptic partners each. Furthermore, any source is restricted to connect to at most $d_s = 16$ ROLLS neurons. Here, the goal is to select those sources which maximally contribute to the postsynaptic firing. At the beginning of the experiment, for displaying purposes, A is initialized to a regular pattern that respects d_s and d_n(see Figure 5.6 A).

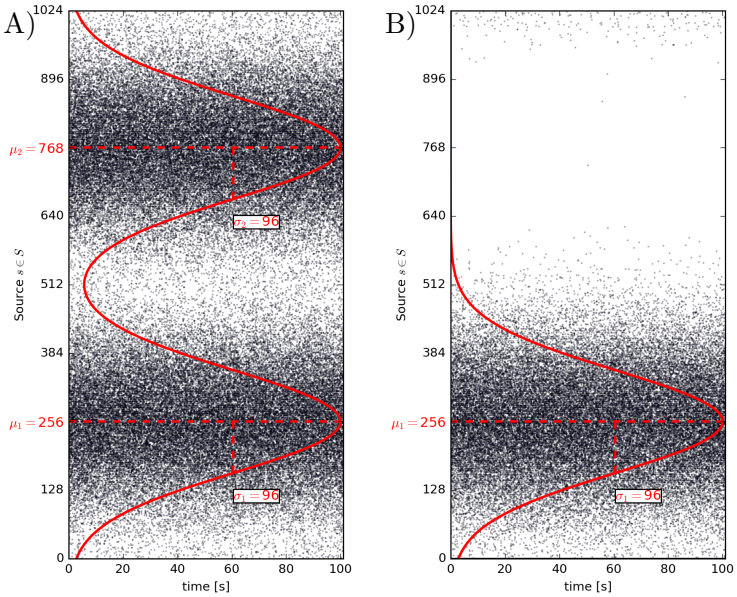


Figure 5.5:
Example raster-plot to illustrate the computer-generated event-stream on 1024 Poisson sources that serves as input to the ROLLS neurons; Red: Distribution of rates. A) Activity of the input population before 'lesioning'. B) Activity simulating lesioning on the source neuron's inputs.

Over the course of 50000 incoming source-events, the network rearranges itself to connect with increased density to those sources that show most activity and neglects those sources which show little activity (see Figure 5.6 B). We performed a lesioning experiment to underline the network's capability to re-arrange under varying stimulus statistics. This may be considered analogous to whisker-trimming experiments in rodents [Holtmaat and Svoboda, 2009] which show an increased restructuring of barrel cortex after the animal is deprived of

sensory input from a number of whiskers. After this first period of the experiment, we removed one of the two modes in the source rate distribution, so that the remaining rates are the largest around $\mu_1 = 256$ with a standard deviation of $\sigma_1 = 96$ (see Figure 5.5). After this change, the continuation of stimulation for another 500000 input events resulted in a successful reconfiguration of A. Here part of the former concentration of connections remains since $d_s = 16$ is for sources of maximal activation around $S = 256$ with the effect that no new connections can be made here. Figure 5.6C shows this reconfiguration as an increase in high-weight connections around μ_1.

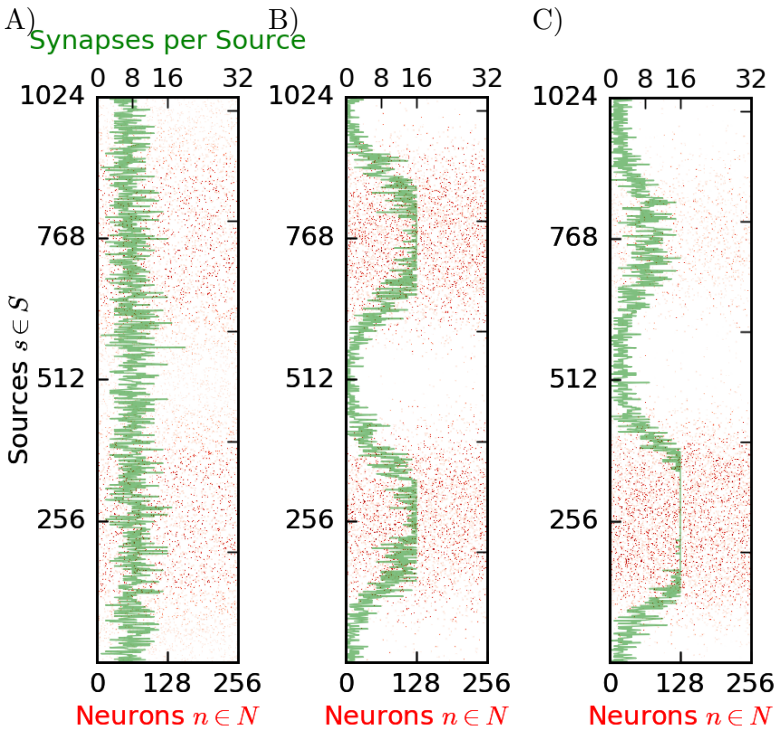


Figure 5.6:
(A) Initial random adjacency, connecting 50% of the sources to the 256 destination neurons, respecting the limitations on fan-out ($d_s = 16$) and fan-in ($d_n = 32$), modified by synaptic plasticity only for comparison. B) Network configuration after presenting 50000 stimuli from 1024 source neurons, spiking with rates distributed as in Figure 5.5A. Note the increase in synapse density, following the presented input Figure 5.5A. C) State of the adjacency matrix after a further 500000 stimuli with changed statistics as shown in Figure 5.5B.

5.2.3 Evaluation of the Compressive Features of Structural Plasticity

Using a matrix that is kept continuously sparse under constrains that can be defined by the user, allows for a dense representation to save memory. In a sense, this can be considered a compression where number of synapses is reduced from a maximum of $\mid W \mid$

$$\mid W \mid = \mid S \mid \times \mid N \mid$$

to at most

$$\mid \hat{W} \mid = \frac{\mid S \mid \times \frac{|N|}{d_s} + \mid N \mid \times \frac{|S|}{d_n}}{2}$$

, respecting the fan-out constraint d_s on the source population S and a fan-in d_n on the destination population N. This allows reserving a fixed number of memory resources for the dense representation of the topology.

The process is not lossless since situations may arise where more than the allowed number of synapses may be assuming a non-zero weight in the uncompressed, full matrix, case. The advantage in the use of structural plasticity as lossy compression of the weight matrix is that the weakest synapses that contribute least to the postsynaptic firing behaviour are those that are pruned first.

Defining the compression ratio C as $C = \frac{|W|}{|\hat{W}|}$, substitution gives:

$$C = \frac{2 d_n d_s}{d_n + d_s}$$

In the implementation, we considered not only the optimization of memory but also the practicality of looking up synaptic connections for routing. Here a trade-off was made,

and the matrix is stored not in its most compact form. Using a dense array allows a reduction of memory used from $\mid S \mid \times \mid N \mid \times \log_2(W_{\max})$ down to

$$d_n \times \mid N \mid \times (\log_2(W_{\max}) + 4 + \log_2(S) + \log_2(ID_{syn}))$$

. In the presented example this is a reduction from $2^{21}Bit$ to $2^{19}Bit$ ($C = 4$). The introduction of structural plasticity to implement this form of memory optimization requires resources on the FPGA. Applying the presented approach of introducing a co-processor into the system to re-arrange the networks topology, comes with an increased the number of occupied slices from 169 to 1840 (26% of available slices) in the used SPARTAN6LX45t FPGA.

5.2.4 Time Analysis

The runtime of the structural plasticity algorithm on the microprocessor is highly dependent on the input stimulus frequency and statistics. For this reason, no scalar value for latency can be determined, since the amount of computation, e.g. to find a potential location for re-instantiation of a synapse, is dependent on the sparseness of the column in the connectivity matrix that corresponds to the postsynaptic locations of the neuron in question.

By far the largest delay that restricts event-rates is that of the lookup and routing of source-events to a targeted destination. This lookup is performed through the matrix W, which is represented in a dense form. This formatting is advantageous in terms of computation on whole columns as it is done in the reconfiguring structural plasticity algorithms. However, when it comes to the determination of stimulus destination, a full search through the matrix has to be performed. For this reason, another matrix A was introduced that serves as

a lookup table and only stores destination neurons, where the sources define the 1024 rows of the matrix. This matrix again takes up a quarter of the original adjacency matrix, so that the advantage in memory is reduced to 50% of the original matrix, when we add A to W.

5.2.5 Simulation of the Biohybrid System

One of the applications of structural plasticity, aside from the advantages in memory use and the increase in scalability through the saving of synaptic resources, is source selection. In the following, we illustrate a neuromorphic system which exhibits structural plasticity for the selection of sources from a simulated MEA. This step was taken to point out a possible application of the present work and is meant to supplement the previously provided main verification experiments.

In the context of a biohybrid system, where the neuromorphic processor is connected to a multi-electrode array via AER and is presented with a high dimensional vector of event sources and a limited number of synapses that can be invested in making a connection to those sources, structural plasticity can be used to select a subset of electrodes for later processing of events.

The 2D-plane of an *in vitro* culture is partitioned in 32×32 electrodes which each have an individual ID. Now events from somata that cover a particular electrode are assigned to its ID to generate address-events. This stream of events can now be sent to the neuromorphic system as an input stimulus.

The most straightforward use of the selected sources could be to treat the biological system like a reservoir which is presented with a stimulus. In this application, the neuromorphic side would perform a classification of the biological network's state. Here, input to the cell culture would be provided via

EOSCs and the readout would be performed by the EOSFET devices. To date, event-based MEA are still subject to ongoing research. Several research groups are working on prototypes of this form of electrophysiological data acquisition and measurement system, as outlined in the introduction of this thesis. The verification of an early MEA prototypes sensor circuit is presented in Appendix A.5. However, since we only had access to a single-channel test-structure at the time, a simulation of an ideal MEA serves as an event source in the following evaluation.

We evaluated the co-processor with activity-dependent structural plasticity in a simulation based on the model of developmental plasticity (see previous Chapter 4). For this purpose, we used the created developmental model on a PC to generate event streams that represent the spontaneous firing behaviour of an in-vitro culture. The use of this approximation of a MEA replaces the real electrophysiological recording to provide a greater degree of controllability. Due to the difference in timescales for the implemented activity-dependent structural plasticity and the developmental model, we did not create a continuous simulation. Rather, the previously outlined developmental model is used to create snapshots of a network topology, and the output adjacency matrix is stored for every day in vitro (DIV). Secondly, a *BRIAN* model was created, using this information. As Figure 5.8B and Figure 5.10B illustrates, this way spontaneous activity causes an oscillatory behaviour that manifests itself through the inhibitory connections of the network. This form of initialization was taken from the *BRIAN* codebase (see *http://briansimulator.org/demo/*).

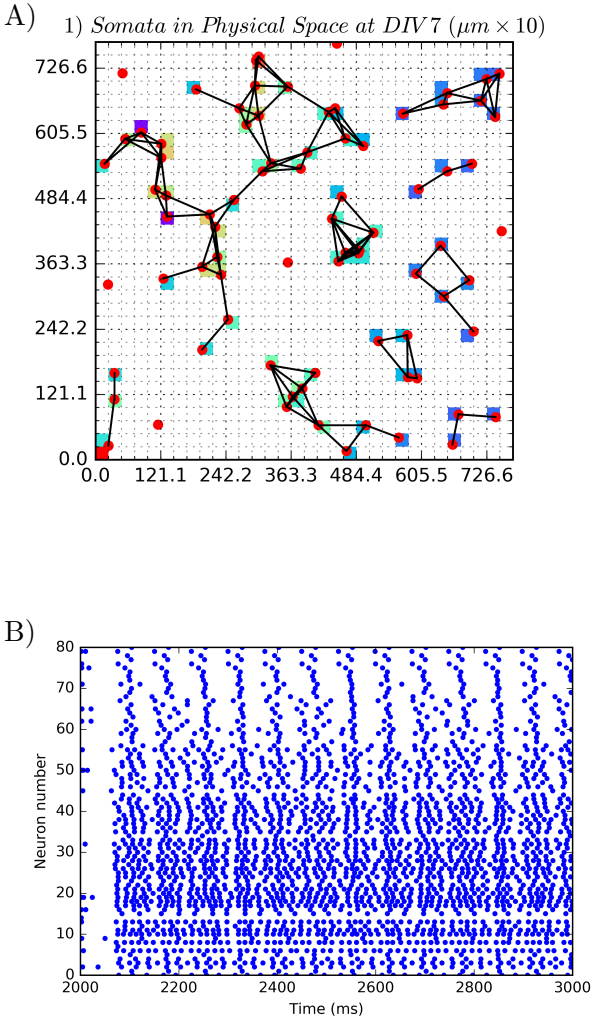


Figure 5.8:
A) Randomly generated cell culture in physical space after DIV7, partitioned into 32×32 *electrodes. B) Raster plot from the underlying network simulation, following the extracted network topology.*

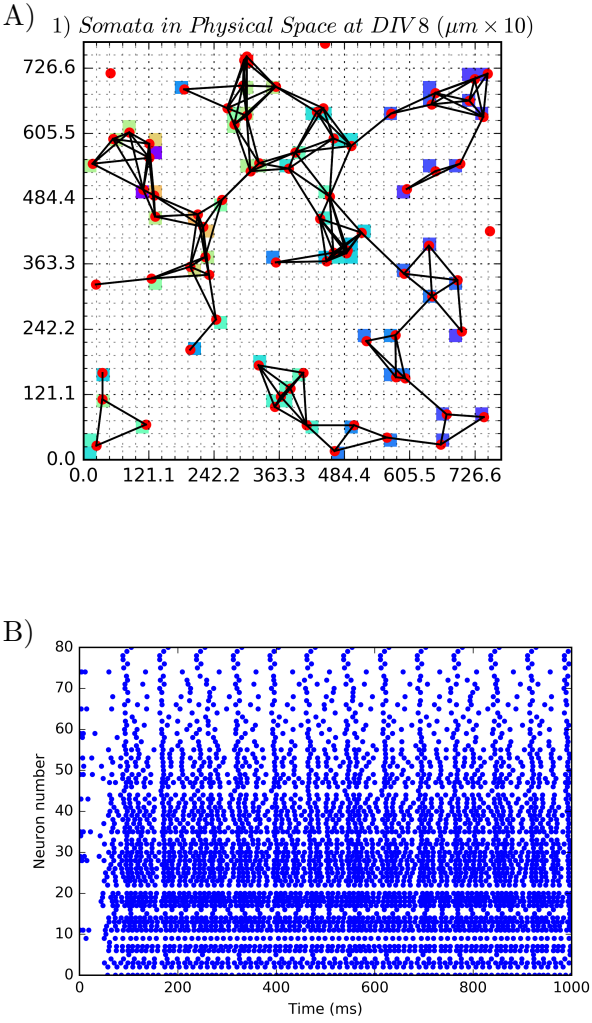


Figure 5.10:
A) Randomly generated cell culture in physical space after DIV8 for contrast. B) Raster plot from the underlying network simulation, following the extracted network topology.

Figures 5.8 and 5.10 illustrate the networks generated for DIV 7 and DIV 8. The background grid shows the partitioning into electrodes and their color-coded mean activity. A more detailed view of the activity of the network is shown as a raster-plot, sorted by neuron ID. We did not consider the electrical characteristics of the extracellular medium, which would cause a co-activation of neighbouring electrodes. Due to the approach taken in the generation of the networks random graph, no morphology of the cells was generated. Therefore, passive membrane properties are not implemented, and the activity of the neurons is local to its soma circuit. The outlined limitations are reflected in the generated event-stream from the simulated electrodes in Figure 5.11 (Left). After logging the generated event-stream, we replayed it to the neuromorphic system centered around the ROLLS through the implemented USB interface. Here, the synapse allocation was performed, and after the stimulation period, the weight-matrix was transmitted back to the PC. Based on this information, Figure 5.11 displays the synapses allocated to each source. Note, that the presented data (Right) is smoothed with a Gaussian filter ($\sigma = 64$). This acts as a qualitative measure of synapse density. Due to the limitation of $d_s = 16$, a total investment of all available synapses to the most active sources is prevented. Instead, these sources receive the largest allowed number of synapses but those sources that see lesser activation are still sampled to a lesser degree.

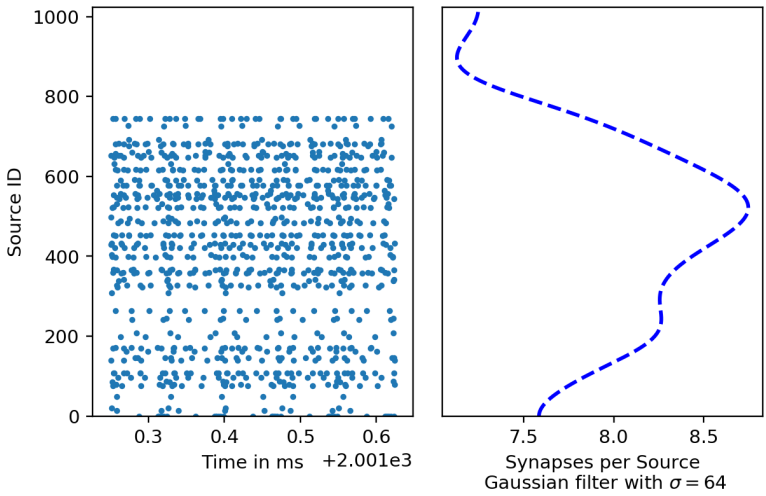


Figure 5.11:
Left: Excerpt from the stimulus raster plot; Right: Number of allocated synapses per source after gaussian smoothing with a standard deviation of $\sigma = 64$. Note that the actual allocation ranges from 0 to 16 synapses per source; The smoothing operation distorts this but was chosen to give a qualitative measure of synapse density.

5.3 Summary

In this chapter, we outlined the approach taken to algorithm design, from the first prototyping phase to the creation and testing of the final algorithm in neuromorphic hardware. In its current state, large parts of the simulation of structural plasticity are executed on the created co-processor. A potential next step in the development of structural plasticity within the created system, an implementation of sub-components directly into VHDL could increase performance by exploiting the ability of parallel computation on a logic level. Along

with the process of algorithm development, the implications of structural plasticity were also explored in conventional hardware and exploited for memory optimization, improving learning speed and the selection of sources from an input vector of events. In the following chapter, we present the overall conclusions of the project and discuss potential and possible future work.

6

Discussion

Although our work on structural plasticity was presented as an application-specific solution to overcome the limited number of synapse circuits in neuromorphic hardware, on a more fundamental level, the rearrangement of network topology may pose a valuable addition to spiking neural networks in general. While we only focused on the selection of sources, [Spiess et al., 2016] and others explore the impact of topology reconfiguration within the network, rather than on its inputs.

As neuron populations in deep neural networks have increased in size, it becomes more and more important to apply memory reduction techniques, even in conventional hardware. Classical compression methods such as Huffman coding have lately found their way into the field of computational neuroscience to reduce the size of weight matrices [Han et al., 2015]. For both neuromorphic implementations and on conventional hardware, we show that structural plasticity can contribute to this effort by allowing control over the degree of sparseness in the network connectivity. Under the aspect of lossy compression, the potential for not only data reduction but also data access reduction is given by the developed biologically inspired model. [Spiess et al., 2016] As outlined in the introductory chapter, the scalability of neuromorphic processors is affected by the area that is necessary to implement a large number of

synapse circuits. Therefore structural plasticity with hard limits on maximum fan-in and fan-out can serve both to compress the networks adjacency matrix through imposing sparseness and to reduce the number of synapse circuits on the neuromorphic processor. Both aspects may increase the scalability of future generations of neuromorphic processors.

The adaptive reconfiguration of network topology is also relevant for existing spiking networks that are designed to solve machine-learning tasks on conventional hardware to not only increase their performance but also lower simulation cost. The implementations of structural plasticity presented provide two modes of operation:

- The number of available synapses could be fixed, to optimize performance on a target platform, e.g., optimization of the use of a limited number of synapses in a neuromorphic processor or reducing simulation time on a conventional computer.

- For off-line training, and where memory resources are available during learning, starting with a denser connectivity and applying pruning during learning, before applying the network to test-data, is possible by manipulating the imposed degree of sparseness over time. This approach was not explored in depth because as we show in Spiess et al. [2016], a constant number of synapses with structural plasticity potentially increases performance while not leading to higher simulation costs after training.

In our developed algorithm, a potential target location for synaptogenesis is chosen on the same postsynaptic neuron that a synapse was pruned from. This choice is performed by locating the maximum weight and taking the center of a normal

distribution from which candidates are drawn such that the probability of locating a new synapse close to a center of activity is larger than that of the use of a more 'remote' location. The notion of space here is arguably vague since we make use of point neurons that do not implement multiple compartments in their neurites or even the cable equations that would model the propagation of potentials within the intracellular medium. Nevertheless, given that the presented sources are structured, and there is a function describing these sources activity as dependent on their identity (or location to maintain the analogy to physical space), the instantiation strategy described leads to faster convergence towards an ideal source selection than a purely random approach. In this process, centers of activity, expressed in large weights, are used as an indicator of where to efficiently place new synapses. As with other parts of the overall algorithm, this allocation could be performed by drawing from globally available information, however, in order not to lose biological plausibility, the chosen constraint in the design of the routine was to operate on locally available information on the postsynaptic neuron.

Throughout the present work, we used static weights on the neuromorphic processors STP synapses and only manipulated a 'virtual' connection weight, as an indicator of connection stability and maturation stage of a synapse. Alternatively, an array of synapses that implement the Fusi learning rule described in Chapter 4 can be used to model LTP in addition to structural plasticity in future work.

A challenge in the translation of biological observations to an algorithmic description of structural plasticity was the use of point-neurons in the neuromorphic processor and the lack of access to postsynaptic current measurements during runtime. Since the observation that both ion dynamics across the cell membrane and carrier dynamics in transistors operating

in the sub-threshold domain are following Boltzmann statistics, which gave rise to the possibility of a direct emulation of biophysical processes in silicon, more than thirty-five years of research in analog VLSI have passed. Since its conception, a trend towards higher levels of abstraction and away from the biological role-model can be observed: Where processors like the ROLLS, used in the present work, still rely on analog sub-threshold neuron circuits, the introduction of AER as a means to create networks using bus communication, forms a design choice that drastically diverges from the point-to-point communication of biological neurons through axons and dendrites. While AER allows overcoming limitations of modern planar fabrication processes, to match three-dimensional neuron cultures regarding connectivity, important biophysical aspects of information propagation between neurons are lost, as digital communication cannot capture the passive membrane properties of neurites. Throughout the presented work we show how these effects contribute to information processing in biological neurons. The diameter of a neurite, distance to the soma and the number of ramifications play a crucial role in the depolarization of neighboring synapses and in shaping postsynaptic currents. While these effects see comparatively little consideration in current research on spiking neural networks, it is intuitive to assume that especially learning would be drastically affected e.g., by the attenuation and delay of sensory input to a local ensemble of neurons, where local connections would cause stronger currents with shorter time-constants. Moreover, local processes of collaboration and competition among synapses, which could be involved in the expression of circuit motifs, cannot be captured. As information on network topology in biological cell cultures becomes more and more available through the rise of connectomics, a morphological neuron model may allow the implementation and analysis of bio-

logically plausible circuit motifs. This approach would allow neuromorphic engineers to profit from neuroscientific research. However, current generations of neuromorphic processors, implement neurons as point processes, so that the direct copying of a biological circuit can be expected to suffer from distorted spike-timings and effective current amplitudes at the soma.

The problem of how to implement such morphological neuron models, which would take up large layout-areas, in aVLSI, is still an open challenge. Solving this challenge would not only increase the degree of biological plausibility in the neuromorphic system but would also allow a greater influence of neuroscience into the field of machine learning and information processing systems.

Alternatively, in a step towards layout area efficiency and away from biological plausibility, structural plasticity could help to re-think current architectures. A dramatic reduction in synapse circuits by a weight dependent switching of synapses might be achievable if each neuron would be interfaced by a bank of synapses that each have a hard-configured synaptic weight. The synaptic weight would be reflected in the synapse ID so that it is possible to change the synapse address as the result of a weight update rule. In this scheme, a weight update would result in a re-routing to a different destination synapse with a different weight. If several sources are connected to the same destination neuron with the same weight, events from these sources would be all routed to the same synapse.

In a neuron with i synapses, that each has a weight $w_i = 2^i$, eight synapses per neuron would suffice to implement the $8bit$ weight resolution, if multisynaptic contacts are used. This scheme would trade off layout-size with throughput on the shared parallel AER bus. Assuming that neurons operate in biological real-time, AER bus-speeds are orders of magnitude faster than the maximum firing frequency of a neuron, and

the AER communication bottleneck would be limited. This is particularly true if the connectivity is kept sparse through structural plasticity and therefore data-rates are reduced.

The presented model of growth processes in an *in vitro* cell culture is not capable of reproducing graph-theoretical measures such as degree-distribution, measured in cell cultures(see [Feldt et al., 2011]). This fact allows the conclusion that mechanisms other than the pure overlapping of neurites are involved in the formation of potential synapses. A crucial mechanism that was not accounted for, may be the expression of adhesion molecules on the cells membranes, as well as the secretion of guidance and growth factors, even within *in vitro* cell cultures. Here, the created simulation of network development over days in vitro served only to form stimuli that were later presented as input to the neuromorphic system in hardware. Instead of the physiological mechanisms of, e.g., axon guidance along guidepost cells and following gradients of factors secreted by the environment of the cell, only proteins in the target cell membrane inform the developing neurite whether a connection is to be established. This process was only approximated by making the formation of an active synaptic connection a stochastic process. Here, this probability derives mainly from the chance of two neurons bringing their axons and dendrites spatially close to one another.

7

Conclusions and Outlook

Part of the conclusions we draw in this section have been published in [Spiess et al., 2016], [George et al., 2015a] and [George and Indiveri, 2016] and are presented in paraphrased form. The scope of the present work was the implementation of structural plasticity in a neuromorphic system. The approach to designing a structural plasticity algorithm started with a review of the literature on biological processes that give rise to synaptogenesis and destabilization of synapses in somatosensory cortex and hippocampus of rodents. We found that a good design for the selection of input sources to a neuromorphic system would be able to model activity driven processes that can be linked to synaptic plasticity. By relying on a mechanism that is comparable to STDP, for describing the process of dendritic spine stabilization, and the introduction of a border condition to this process which triggers synapse destabilization and pruning, a computationally efficient approach to the chosen problem was formulated.

The collected biological observations and their effects were interpreted to find hypothetical computational primitives that are translatable into the target neuromorphic system. In this way it was possible to create a system that exhibits a certain degree of biological plausibility while profiting from a resource efficiency, that would be hard to match with a more naive approach that could, e.g., operate on global information,

rather to the information that is locally available in the post-synaptic neuron. Likewise, biological observations inspired, for instance, the implementation of weight normalization as a subtractive rule. Our abstraction of biological observations to algorithm design is however based on unverified hypotheses that we formed but did not evaluate experimentally, primarily because our focus was on solving the chosen problem of source selection, rather than advancing the state of research in neurobiology. Nevertheless, throughout the present document, we point out directions for future experiments in this field and provide a neuromorphic platform for testing new hypotheses on the computational primitives behind observations in neurobiology.

To be able to execute the algorithms described in Chapter 4 in hardware, the developed neuromorphic system, created around the ROLLS neuromorphic processor serves as a platform for both routing events and reconfiguring network topology.

By creating a programmable system on chip (pSoC), we provide a platform that allows the fast and easy prototyping of novel event pre-processing and post-processing schemes, as well as the evaluation of novel learning rules. Although our main motivation behind the digital co-processor we created, was its application in simulating structural plasticity, we illustrate the applicability of the programmable system-on-chip in several projects, where tasks such as the precise generation and control of stimuli allows an efficient characterization of circuits within the neuromorphic processor(see Appendix A.3). Moreover, event-filtering algorithms and real-time processing were shown on DVS input (see Chapter 3) In future work, e.g., effects of hormonal neuromodulation, e.g., through dopamine, in reinforcement learning scenarios, may be explored through a continuous modification of biases by

the developed co-processor, according to a model that could be formulated in the *C* programming language. In previous systems that use the monsterboard setup described in Chapter 3 Figure 3.5, FPGAs have been used mainly for providing an interface to conventional computers. The programmable System-on-Chip is an addition that increases the flexibility of the system in that it allows the standalone operation without a host computer. Especially in robotic platforms, initialization of neuromorphic processors and the conversion from an event-based output to the control of actuators could be performed by the pSoC. The ability to program the processor in *C* saves users having to develop digital hardware designs in VHDL. As a trade-off against the resulting flexibility, in a highly optimized application, the processor's routines can be replaced with dedicated digital hardware peripherals that communicate with the co-processor via the created bus architecture for application specific extensions of the capabilities of the platform.

Our verification of the structural plasticity algorithm on the pSoC created illustrates that it is possible to perform the involved computations in real-time, as the neuromorphic system receives external stimuli. With this work, we implement a model that operates on the same time domain as synaptic plasticity, yet gives rise to slow dynamics of topology reconfiguration, without the necessity of having to introduce a second time domain with time-steps that are large multiples of those of the synaptic plasticity time domain. This approach is made possible through our extensive use of interrupt routines, and increases the biological plausibility of the system.

The development of the algorithmic implementation of structural plasticity was performed gradually, starting from a proof of concept in *BRIAN* and *Python* (See Appendix A.1). Early experiments on conventional hardware formed the basis for a

more extensive application in spiking neural networks. At this stage, different network topologies were used to evaluate the performance of a group of derived algorithms which gave rise to the work presented in [Spiess et al., 2016].

The simulations performed here highlighted that the noise in the network's representation of an input is reduced roughly 30% faster with structural plasticity. This reduction of noise in the responses means that the networks can transmit the represented input with a clearer signal to connected populations. As a less noisy input representation is propagated through the network, learning speed is increased.

Besides the functional properties reported in Spiess et al. [2016], the aspect of structural plasticity that we investigated in the present document, is the optimization of the use of memory resources as discussed in Section 5.2.3. The developed algorithm opens novel, biologically inspired perspectives on resource optimization in neuromorphic hardware and promises a solution to scalability issues by eliminating unnecessary synapses and by exploiting sparse connectivity for a compression of adjacency matrices.

The use of structural plasticity in a compression scheme relies on fan-out and fan-in constraints. A future subject of research that is based on the algorithm introduced here, could be the creation of constraints that instead impose a degree distribution within the network. Use of such more sophisticated constraints may lead to the emergence of re-occurring circuit motifs in the functional connectivity. For instance, with an introduction of spatial dimensions in the network, the probability of introducing connections close to the soma could be increased for neighboring neurons, which would lead to an increase in local excitatory connectivity, one of the prerequisites for winner-take-all like motifs.

The created model of structural plasticity introduces the no-

tion of 'silent' synapses that only hold NMDA receptors and can only be co-activated by mature neighbors on the same dendritic branch. This distinction from mature synapses comes with the benefit of reducing the number of weight updates through synaptic plasticity and a reduction in AER throughput since silent connections are not used for event routing. Mature synapses have an STDP-like behaviour that describes not their electrophysiological properties but their stability and degree of maturation as described in Chapter 4. Mechanisms of local competition through weight normalization and collaboration through co-activation among synapses, allowed to implement the formation of clusters of synapses.

In both the version that is aimed at neuromorphic hardware and in the version that targets conventional hardware, we begin training on a randomly initialized sparse connectivity-matrix. Whenever a connection is destabilized and pruned, another connection is made to the same postsynaptic neuron. This allows the use of a fixed number of synapse circuits in hardware and allows to maintain a fixed sparseness in the adjacency matrix which reduces memory utilization and can be considered a form of lossy compression.

In the second stage of the project, after algorithm prototyping and optimization, we migrated the algorithm into the neuromorphic system. To increase the number of inputs that the neuromorphic processor can operate on, structural plasticity is used as a source selection mechanism. Here, the behaviour of the network under changing input statistics was investigated. We show, that the developed solution is capable of forming clusters of connections around those sources that contribute most to the firing of the postsynaptic destination neurons.

Future developments will explore the organization of structurally plastic recurrent neural networks in an unsupervised learning experiment, to investigate the impact of the devel-

oped algorithm on network dynamics.

Structurally plastic neuromorphic biohybrid systems may be used in future work, to treat biological cultures as a reservoir network that receives input stimuli through the co-processor. Here, the neuromorphic hardware would perform a source selection among the MEAs electrodes and a subsequent classification of the culture's firing behaviour. This use of the biological subsystem in the hybrid network would profit from homeostatic mechanisms and adaptation in the biological culture, which stabilize its network dynamics. Besides using the biological side for computation, stimulation and modulation schemes may be used e.g., for the disruption of pathological network behaviour such as epileptic seizures, in a closed-loop scenario. Here, the reduction of sources to those that are most informative ones would decrease bandwidth requirements and may improve the system's ability to react to seizure onsets in real-time, by targeted stimulation.

In its present state, the presented work influenced several works, both in the field of neuromorphic engineering [Roy and Basu, 2016] and in the field of computational neuroscience [Deger et al., 2016].

A

Appendix

A.1 PC-based Prototyping in Python

In this first implementation of the previously described routines, the python based spiking neural network simulator *BRIAN* was used to prototype and develop the general structure of the algorithm and to perform the first verification of the concept. The reason for opting for this platform was the flexibility and the large number of functions that are available in *Python*, which allowed the rapid evaluation of different design choices.

In this first simulation, structural plasticity was tested in a small network with two interconnected cultures, with ten neurons each. The simulation is started with 30% of the 100 potential synaptic connections of the 10×10 adjacency matrix between the two layers were initialized as mature synapses. The remaining elements of the matrix are left empty, signifying the absence of a synapse. Stimulation of the neurons was performed by a constant direct current stimulation to the pre- and postsynaptic neurons in a precisely timed offset of stimulus onset between the cultures, so that the neurons of both cultures always fire in a fixed phase offset to each other and with a fixed frequency (the neuron model below shows that no stochastic elements are implemented). The implemented neuron equation is as follows:

$$\frac{dV}{dt} = \frac{(g_e \times (E_e - V_r) + R_m \times I + (E_l - v))}{\tau_m} \tag{A.1}$$

$$\frac{dg_e}{dt} = \frac{-g_e}{\tau_e} \tag{A.2}$$

Here the discrepancy in the units in the first differential equation was neglected. The synapses between the involved neurons follow the following plasticity rule for their weight updates:

$$\Delta W_{i,j}(s) = \begin{cases} dA_{pre} \cdot e^{-s/\tau_{pre}}, & \text{if } s > 0 \\ -dA_{post} \cdot e^{s/\tau_{post}}, & \text{if } s < 0 \end{cases} \tag{A.3}$$

BRIAN's support for running user-defined routines periodically during the run-time of the network simulation, was used to implement structural plasticity. In this implementation, the matrix T implements a dedicated timer for every synapse where the elements $t_{ij} \in T$ represent the iterators. On every execution, all iterators are incremented where synapses are under test. In this version of the algorithm, T implements the previously described destabilization time window in which a synapse is unstable. In the event of a synapse reaching the end of the time-window and its weight being below the threshold θ, pruning occurs and the timer t_{ij} is reset.

As the simulation is executed, Algorithm9 is triggered periodically, every ten iterations of the simulation.

Algorithm 9: Network operation procedure. $synapse.W$ denotes the object attribute W of the object *synapse*

Data: *BRIAN* connection object *synapses*
Result: *BRIAN* connection object *synapses*

$W \leftarrow synapses.W \times used$ /* extract weight matrix from connection object */
$T[w_{ij} \leq \theta] += 1;\ w_{ij} \in W$ /* update timers */
$T[w_{ij} > \theta] = 0$
$T[w_{ij} == 0] = 0$
$kill =| T[T_{ij} > t_{prune}] |$ /* detect timeout in T, prune accordingly */
$wbuff[T_{ij} > t_{prune}] = 0$
$T[T_{ij} > t_{prune}] = 0$
$blacklist[T_{ij} > t_{prune}] = 1$
$in\_use = blacklist + wbuff$
$empty =| in\_use[in\_use == 0] |$ /* find empty spots */
if $kill < empty$ **then**
    /* if we have enough space to replace all synapses */
    for $k = 0$ *to* $| kill | -1$ **do**
        $new\_x = empty[0][rand(0, empty)]$
        $new\_y = empty[1][rand(0, empty)]$
        $wbuff[new\_x, new\_y] = theta - 0.00001$
        $used[new\_x, new\_y] = 1$
else
    for $k = 0$ *to* $empty - 1$ **do**
        $new\_x = empty[0][rand(0, empty - 1)]$
        $new\_y = empty[1][rand(0, empty - 1)]$
        $wbuff[new\_x, new\_y] = theta - 0.00001$
        $used[new\_x, new\_y] = 1$
$synapses.W \leftarrow wbuff$ /* write back to connection object */

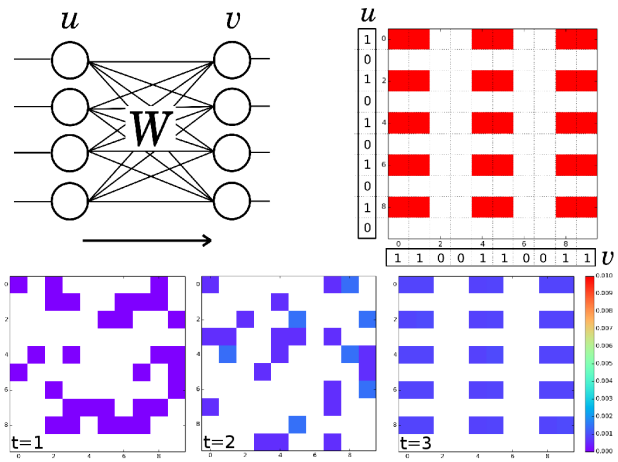


Figure A.1:
Simulation of the outlined structural plasticity algorithm on a network of 2×10 *neurons in a simple associative memory task.*

The algorithm outlined here is the minimal working example of activity dependent structural plasticity following the group of broad concepts described. Figure A.1 visualizes the performance from initialization ($t = 1$) to the final arrangement of the synapses ($t = 3$), compared to the ideal (top right).

Notable here is that at this stage, re-instantiation of synapses after pruning is happening globally, in a uniformly distributed fashion and not only on the postsynaptic neuron, that previously saw pruning of one of its synapses. Furthermore, competition mechanisms, which will provide a method for normalization in later implementations, are not included. Nevertheless, the core concept of the introduction of an edge-case to synaptic plasticity as a strategy for implementing activity-dependent plasticity is present. With this starting point in mind, key functions were re-implemented in C, specifically de-

signed for the task at hand. With the aim of verification of the taken approach, a structurally plastic network topology with sparsity constraints was compared to a network with a full adjacency matrix and STDP only, where synapses can assume a zero-weight state. This state was interpreted as the functional equivalent to the absence of the synapse in the case of a sparse matrix. This comparison was simulated with the same culture sizes as above. At run-time, pre- and postsynaptic activity gradually shapes the sparse adjacency matrix that connects one group of neurons to the other, and over time, the adjacency matrix should become increasingly similar to that of a fully connected adjacency matrix. Whereas some weights remain at zero-value and others become maximally potentiated in the case of the full adjacency matrix, under sparsity constraints, structural plasticity causes the pruning of zero-weights and the re-instantiation of connections at locations where the full adjacency matrix also has nonzero values.

In order to show that both simulations converge towards the same weight matrix when presented with the same stimuli, the Euclidean distance between the two weight matrices is used as a performance metric and computed with every structural plasticity operation during evaluation. The result is illustrated in figure A.2.

This work formed the starting point for a family of versions of structural plasticity later extended by [Spiess et al., 2016] to target simulations of spiking neural networks on conventional hardware.

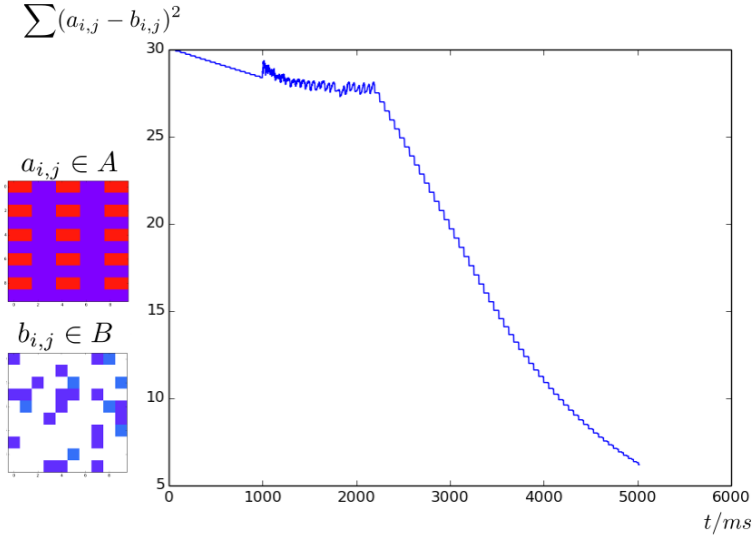


Figure A.2:
Comparison of the simulated small-scale network with a network of same dimensions but only performing STDP on a dense (fully populated) weight matrix.

A.2 Weight Homeostasis, Algorithmic Description

The following routine has the purpose of performing a subtractive form of weight normalization on the connection's neighbour, a contact of which has seen a weight update through the synaptic plasticity routine. For this purpose, the postsynaptic neurons address, as well as the value Δw of the weight update and its sign are passed to the routine. (See Algorithm 10)

Algorithm 10: Weight Homeostasis

```
   Data: neuron, delta_w, selector
   Result: void
   /* selector = 1 → LTD on minima                    */
   /* selector = 0 → LTP on maxima                    */
 1 abs_ampa = abs(delta_w)
 2 rand_seed = rand(·)
 3 if selector == 1 then
 4     θ = W_init
 5     while slot < abs_ampa
 6         for i = 0 to d_n − 1
 7             synapse =
                find_synapse((i + rand_seed)%64, neuron)
 8             if synapse.w = θ
 9                 W_places[slot] = (i + rand_seed)%64
10                 slot++
11             if slot = abs_ampa
12                 break
13         θ++
14         if θ > maxW
15             break
```

```
16 else
17     θ = last_Wmax[neuron]
18     while slot < abs_ampa
19         for i = 0 to d_n − 1
20             synapse =
                find_synapse((i + rand_seed)%64, neuron)
21             if (synapse.w == θ) ∧ (synapse.state ==
                mature)
22                 W_places[slot] = (i + rand_seed)%64
23                 slot + +
24             if slot = abs_ampa
25                 break
26         θ − −
27         if θ = 0
28             break
29 if (selector = 1) ∧ (delta_w ≠ 0) then
30     for ampacnt = slot down to 0
31         synapse =
            read_post_neuron(W_places[ampacnt −
            1], neuron)
32         if synapse.state = mature
33             synapse.w − −
34             matrix W ← synapse
35             if synapse.w = 0
36                 synapse.state = silent
37                 update timeout_sr← (neuron, synapse)
38                 matrix W ← synapse
39                 silent_tot + +
```

```
40 else if (selector = 0) ∧ (delta_w ≠ 0) then
41     for ampacnt = slot down to 0
42         synapse =
            read_post_neuron(W_places[ampacnt −
            1], neuron)
43         if synapse.state = mature
44             if synapse.w ≤ maxW then
45                 synapse.w = maxW
46             else
47                 synapse.w + +
48             matrix W ← synapse
```

A.3 Evaluation of Stochasticity in Neuromorphic Synapses

Besides the number of synapses, another factor that limits the scalability of neuromorphic hardware is the implementation of the memory of synaptic weights. Neuromorphic chips like the ROLLS processor implement weights as a number of current sources which sink charges into the neuron's soma once activated. The amplitude of the postsynaptic current generated like this is the sum of these source currents, which are proportional to bias gate-voltages.

Encoding a synaptic weight as a sum of currents results in a quantization of the synaptic weight to a few discrete states. As well as the weight, which defines the amplitude of the post-

synaptic current, the duration of the signal is also limited in its resolution. Here, mismatch due to layout tolerances affects the pulse-extension stage in a way that variability between synapses is created in a temporally static manner, comparable to fixed-pattern noise in image sensors.

Synapses in biological systems face a different effect that results in a comparable reduction of weight states: synaptic transmission is shown to be non-deterministic due to temperature dependent effects and the previously discussed Brownian motion of receptors in the postsynaptic density of a synapse. Here, this results in a noise component which affects amplitude and duration of the EPSC and defines the number of distinguishable states. To alleviate this precision limiting factor two strategies have been explored in the past. On the one hand, the placement of multiple synapses between a pair of neurons or an input source and a neuron allows averaging out spatial mismatch effects in a biologically plausible manner. Here structural plasticity can extend synaptic plasticity to control the overall weight of the contact [Deger et al., 2016]. The splitting of input sources across several synapses may prove a better alternative, as long as the computational overhead is manageable. On the other hand, an alternative approach is to gain control over mismatch effects instead of compensating them. Balancing amplitude and duration of the EPSC through bias parameters can be used here to maintain a constant area under the EPSCs curve while manipulating the inter-synapse variability of the waveform since length is more vulnerable to mismatch than amplitude. This compensation strategy affects the firing of the neuron if several statistically independent inputs are present, since their resulting EPSCs overlap to lesser or larger degrees, depending on the length parameter.

As a step in the development of the necessary software and hardware framework for running activity dependent structural

plasticity algorithms in the neuromorphic system, the experiment described in the following aimed at characterizing the ROLLS neuromorphic processors synapses with the prospect of finding methods of balancing biases for shaping postsynaptic potentials as described above. In future applications, this characterization might be used in order to encode morphology-dependent effects on the EPSC, by, e.g. selecting synapses with longer time-constants and smaller EPSC amplitudes for distal connections.

As with the strategy pursued during the pSoC development phase, synapse characterization was also chosen as a toy problem to develop an environment for the ROLLS neuromorphic processor and to gain insights into its operation that guided the software design. In the context of the present work, this section describes the circuit and layout of the synapses used.

A particular focus in the synapse characterization was put on the development of a repository of VHDL IP cores and C-libraries to interface and program the ROLLS chip from the soft processor on the FPGA and also on to the creation of user interfaces for stimulation and analysis on the host computer.

Parts of the following description of the characterization experiments were taken from George and Indiveri [2016]

Device mismatch is a phenomenon that affects transistors in different ways, depending on their operating domain. In particular, transistors operated in the sub-threshold domain have significantly larger mismatch than transistors operated above threshold [Pavasović et al., 1994] [Serrano-Gotarredona and Linares-Barranco, 1999]. To evaluate to what degree balancing EPSC parameters to affect variability in firing response is a feasible solution to control device mismatch effects, the main focus was put on the pulse-extension stage of the synapse circuits in the ROLLS chip. Figure A.3 displays a schematic of the circuit involved that generates the input signal to the first-

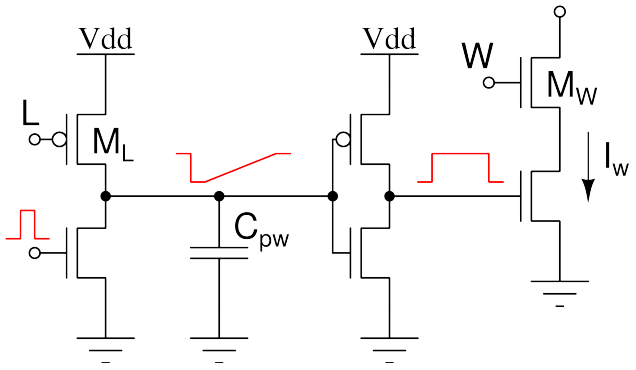


Figure A.3:
Circuit diagram of a synapse pulse extender circuit. Fast input address-event pulses are fed into a starved inverter, which converts them into a waveform that is first reset and slowly recovers. The length of this recovery, and of the digital pulse generated by the subsequent inverter is controlled by the voltage Bias L. The output of the synapse is represented by a current I_W whose amplitude depends on the bias voltage W. Output currents of multiple synapses can be summed together and fed into neuromorphic circuits to implement synapse and neural dynamics.

order linear low-pass filter that acts as an excitatory synapse-emulating aVLSI circuit.

Setting a sub-threshold bias for M_L will produce mismatched currents that will integrate their differences over time to reset the input pulse, leading to a large amount of variability in the net effect of the synapses biased in this way. Conversely, by operating M_L above threshold, (and compensating the bias of M_W to maintain the net synaptic drive), the neurons driven by these synapses will show much less variability.

Stimulation via the AER protocol is performed automatically using the soft processor in the system, following a predefined protocol that specifies the type, target location, and

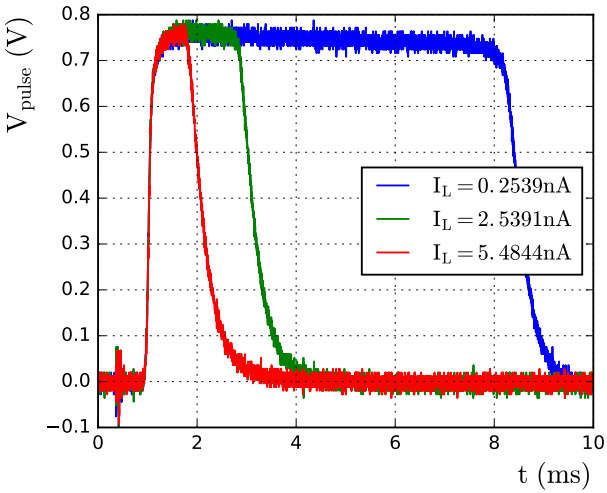


Figure A.4:
Voltage output of the synapse circuit measured in response to pulses with three different lengths, for a constant weight bias setting $I_W = 0.125\,\mu A$.

frequency of the input stimulus. To allow for fast characterizations, the focus of the experiments described is on the recording of the digital spikes transmitted to the FPGA rather than a direct investigation of the EPSCs. This allows fast characterization and increases the usability of the tool-chain in everyday research applications. To evaluate the precise timing of the neuron spikes, we time-stamped the spikes as they were being measured by the FPGA device, using a timer with 40 ns resolution, and transmitted the address and time-stamp of the measured event to a host PC for logging data via the FX2 USB link. All further analysis is performed on the data logged on the host PC and verified manually on the oscilloscope by directly observing V_{mem} and V_{pulse}.

To determine to what extent it is possible to modulate the effect of device mismatch in the ROLLS neuromorphic processor, we stimulated one synapse per neuron and measured the variability of the response of different neurons. As mentioned in Section 3.1.1, the current produced by the mixed-mode pulse extender synapse is integrated by a first-order current mode linear filter. The final synaptic current sent to the neuron can be described by:

$$\tau \frac{d}{dt} I_{syn} + I_{syn} = \alpha I_W \tag{A.4}$$

where τ is a time constant directly proportional to the filter circuit capacitance and inversely proportional to a user programmable bias current. Similarly, α is a gain term, tunable by additional user programmable bias currents (see Qiao et al. [2015] for a detailed explanation of the circuit details). The change in synaptic current ΔI_{syn} is therefore directly proportional to I_w and to the pulse duration ΔT ($\Delta I_{syn} \propto I_w \cdot \Delta T$). However, while I_W can be directly programmed by the on-chip current-mode bias generator (which produces the corresponding voltage bias W to apply to M_W of Figure A.3), the pulse duration ΔT can only be controlled indirectly, by programming different values of I_L (i.e., the current flowing through M_L of Figure A.3). In the particular implementation of the proposed circuit in the ROLLS chip, a capacitor C_{pw} of 500 fF was used.

In Figure A.4 we plot the voltage output of the log-domain pulse integrator in response to pulses extended by different values of I_L. As expected, smaller bias currents produce longer pulse durations. To show that longer pulse durations effectively increase the total synaptic current, we also plot the neuron's response to these pulses, for a constant I_W bias, in Figure A.5. As the total synaptic current integrated by the sil-

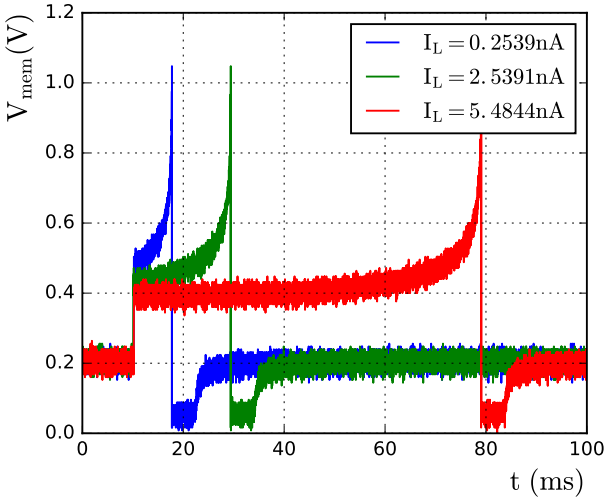


Figure A.5:
Silicon neuron membrane potential measured in response to the three pulses plotted in Figure A.4, and for a constant weight bias setting $I_W = 0.125\,\mu A$.

icon neuron circuit is larger for longer pulse durations, smaller settings of I_L produce output spikes earlier: neuron response times to input pulses with large ΔT are shorter than response times to input pulses with short ΔT.

So as to collect statistics and perform large sets of quantitative measurements on the variability of the neuron's response times, we interfaced the ROLLS chip to an FPGA device and stimulated it using the AER communication protocol. We measured the time-stamps of the address-events both on the input to the ROLLS (to stimulate the synapses) and on the output (to measure the neuron response times). Relative timestamps of 40 ns resolution were attached to address-events produced by the chip, to evaluate the precise timing of the response-spikes. All communication and routing of events is managed in real-time by a soft processor implemented within the programmable logic fabric of the FPGA. The FPGA device then transmits the data to a host PC, via a USB link, for further processing and analysis.

After the initialization of the ROLLS neuromorphic processor circuits with a proper set of analog biases, we configured the setup to stimulate a single synapse per neuron, in all 256 neurons in parallel. This can be achieved by broadcasting a single address-event across all rows of the synaptic array on the chip. We configured the synapses and neurons to produce an output spike in response to a single input address-event. At the onset of the experiment, the FPGA resets its time-stamp generator and subsequently transmits the input address-event to the ROLLS chip. Neuron response times are measured in a time window of 1.5 s, and corresponding pairs of output neuron address and time-stamp relative to stimulation onset are measured. The experiment is repeated for different values of I_L ranging from 0.25 nA to 5.99 nA. To compensate for the change in pulse duration, we also adjust the value of I_W such that the

neuron average response time lies around 31 ms although the pulse-width changes with I_L.

Figure A.6 shows the standard deviation of the spiking output time of the neuron, in response to a single synaptic input, measured across the population of neurons on the chip, and repeated for different values of the leak current I_L, which is inversely proportional to the synaptic input pulse width. As expected, larger pulse widths (smaller I_L settings) increase the variability across neuron responses, while smaller pulse widths give rise to less variability, despite the fact that the neuron response times are approximately the same (thanks to the larger synaptic weight settings I_W used to compensate for the shorter pulse widths).

Figure A.7 shows the same measurement made for all 256 synapses, for the bias setting $I_W = 0.33\,\mu A$ and $I_L = 2.01\,nA$. The proposed balancing of biases on minimum-size transistors for time-dependent signals with the use of larger geometries for transistors that control the synaptic weights effectively controls stochasticity in the EPSCs provided to the postsynaptic soma. We characterized how the variability can be modulated by changing circuit configuration and biases while maintaining the neuron's overall response properties constant. The set of tools developed and used in this work, including the FPGA setup, provides a fast and reliable infrastructure for quantifying the variability present in the device under test through the measurement of the spike response times in response to single stimuli. We can now use this framework to investigate how the tunable variability of the neuron response times in the neuromorphic processor can be used to implement efficient random forest and bagging techniques for improving their classification accuracy, and to give rise to stochastic behaviours in recurrent networks for implementing probabilistic neural models of computation and building networks able to carry out inference on

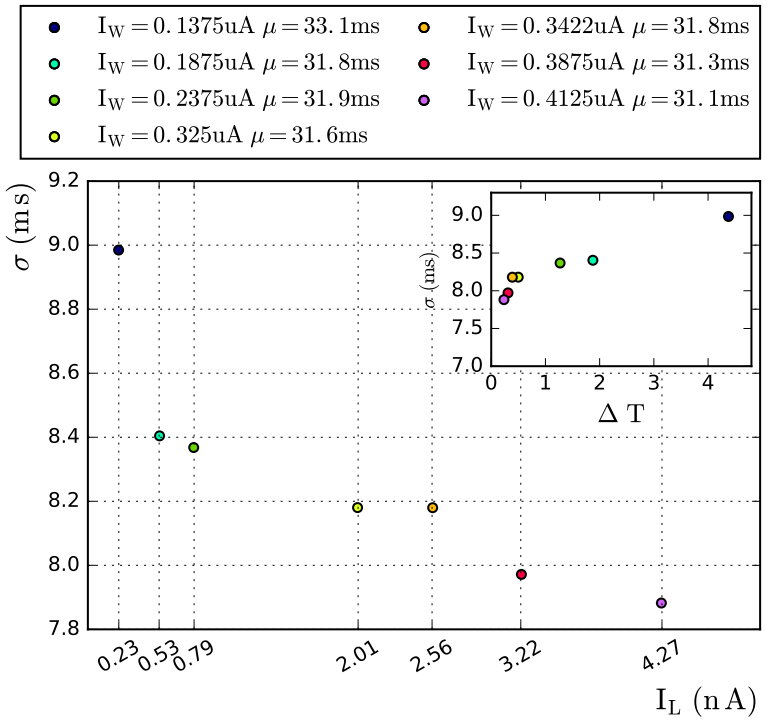


Figure A.6:
Standard Deviation σ of the neuron response times measured across the 256 neurons on the chip, for different settings of I_L bias. Note that the neuron response times are kept approximately constant, by compensating for the changes in pulse duration with changes in synaptic weights. Inset is the same data, plotted over $\frac{1}{I_L}$ as it is proportional to ΔT.

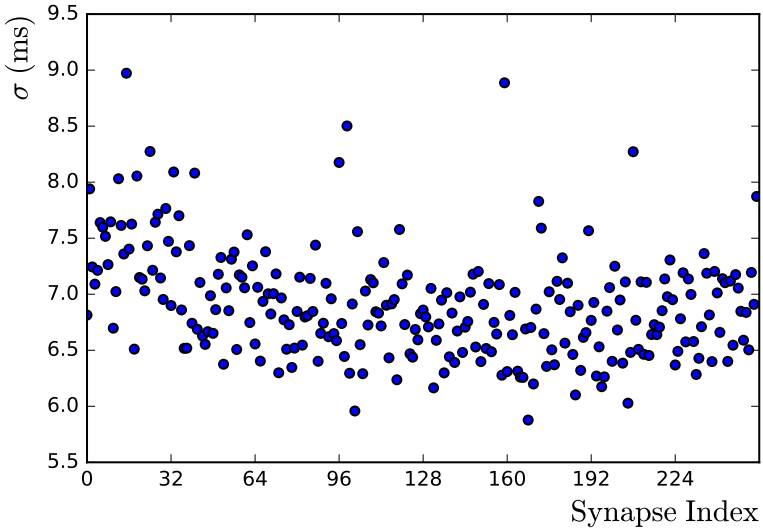


Figure A.7:
Standard deviation of spike times ΔT produced by the population of neurons with $I_L = 2.01\,nA$, measured across the 256 synapses afferent to each neuron.

their input data.

A.4 PCB Designs utilizing the described pSoC

Throughout our work, several chip platforms were designed for specific applications. All these systems use the Raggedstone 2 board as a motherboard so that the implementation of structural plasticity using the developed co-processor is possible on all of the platforms developed as described in the following section.

A.4.1 The REX Platform

The REX board (Figure A.8) interfaces the Raggedstone 2 board through multi-pin connectors. This board was designed in collaboration with Marc Osswald and holds the voltage converters that are needed for peripheral modules that can be connected with it in a modular fashion. Furthermore, with the help of port-expanders, the 136 available I/O pins are expanded to 128 slow pins, controllable via I^2C that are used for the initial configuration of the neuromorphic hardware and 132 fast pins that are directly connected to the Raggedstone. These fast pins are used for high-throughput communication via parallel AER for the interfacing of neuromorphic hardware and for the USB interface module that provides a bidirectional connection to the host PC. The neuromorphic IC sits on a PCB that takes one of the four free slots in the system. A second slot is occupied by the USB interface module. In future experiments, the remaining slots will be taken by either additional neuromorphic ICs or interfaces to electrode-arrays

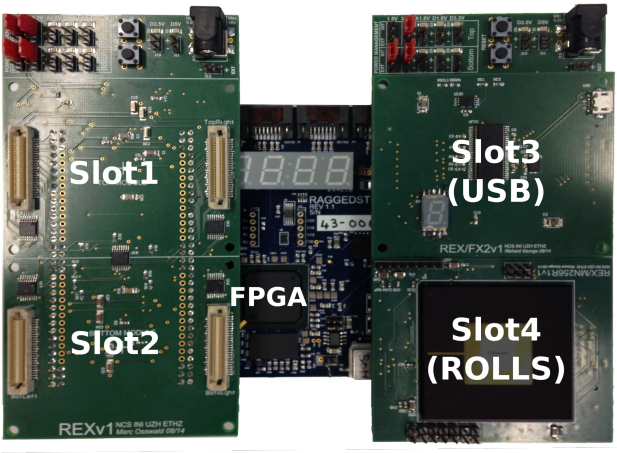


Figure A.8:
The REX setup, created in collaboration with Marc Osswald. The FPGA boards I/O pins are used to connect two REX mainboards, each providing two sockets for ROLLS chips

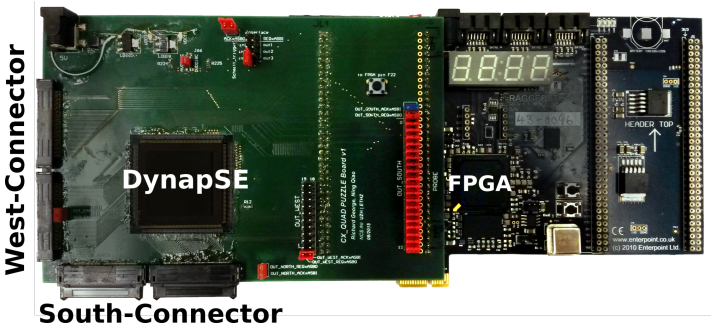


Figure A.9: *The PUZZLE platform, in single chip configuration.*

for recording from, and stimulating neuronal cell cultures.

The top-right slot in figure A.8 is taken by a USB module which is functionally a clone of the solution developed by *D.Fasnacht*.

A.4.2 The PUZZLE Platform

The PUZZLE platform forms an interface to another neuromorphic processor, the *Dynapse* developed by *Qiao et al.*. The configuration presented in figure A.9 shows only a single-chip setup. However, the system can be extended with up to 15 daughter-board tiles, to form a cluster of 4×4 processors. This connection is established through the south and west connectors to form a larger plane of PCBs. (See Figure A.9)

A.4.3 The RAMP Platform as a Neuromorphic Memristor-Driving System

We designed a system around the RAMP-Chip as a device for driving memristive devices in a crossbar configuration. (Fig-

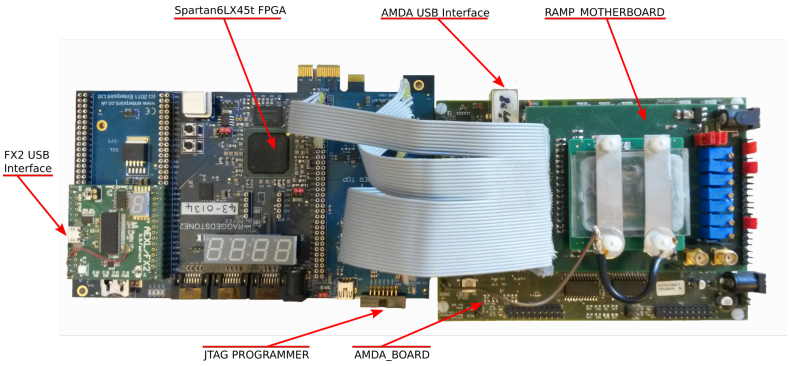


Figure A.10:
Experimental setup directly connecting the RAMP IC to external memristors. Here the chip itself is covered by a protective hard-plastic case

ure A.10) This system created around the RAMP chip consists mainly of three components:

- The *RAMP_MOTHERBOARD* supplies power to the chip and allows a lowest-level configuration of the chip by jumper settings. This PCB also serves as a break-out board that allows users to connect memristors and electrophysiological measurement equipment to the chip and to monitor waveforms such as membrane voltages.
- A DAC bank (AMDA board) which supplies bias voltages to the chip to provide the integrated circuits with parameters such as firing thresholds and time constants. The AMDA board is interfaced via USB.
- An FPGA board for digital communication, e.g., to transmit monitor and stimulus commands. Here a Cypress Fx2-based module provides a second USB interface to a PC.

These system components are depicted in figure A.10

A switch PCB between the chip and the memristive device is used to allow automatic switching of the memristor between the automated test equipment which is used to carry out DC operations on the device (forming, voltage sweeps to set the device resistance to a well-defined value, read the device resistance), and the neuromorphic chip. The switch PCB has the sole purpose of allowing the characterization of the system without damaging the device. It is therefore fundamental in the initial phase of the experiments, but it does not affect the validity of the results.

The neurons implemented in the IC and driving the memristors are based on an exponential leaky integrate and fire model. In each neuron block, we used a neuron circuit based on the adaptive exponential integrate and fire neuron described in Qiao et al. [2015]. The plasticity model used here is based on the spike-based perceptron learning algorithm of Brader et al. [2007]. In this model, the relevant neuron states are $\boldsymbol{s}(t) \coloneqq (V_{mem}(t), C(t))$ which are the neuron's membrane potential and the intracellular calcium concentration respectively. The latter is a low-pass filtered version of the neuron's spike train. The IC itself was directly bonded to a carrier PCB and connected to the breakout board via spring-loaded connectors.

To operate these circuits that allow a biologically inspired potentiation and depression of memristors that serve as memory elements in the synaptic interconnection of neuromorphic models of cortical neurons, on the PC side, drivers for both the bias-setting AMDA Board and the digital part of the system are used by the graphical user interface (GUI). This software provides a single interface for loading and defining bias voltages, and stimulating and monitoring the neurons.

To allow a convenient and easy use of the setup, a GUI was implemented in PyQT and Python, to communicate both

with the AMDA board and the FPGA through the two USB interfaces. From this single environment, configurations can be created, saved to, and loaded from a file so as to control the chip and use it in memristor interfacing. Furthermore, stimulation of the analog sub-threshold neuron circuits can be performed by specifying target ID, frequency, and duration of the stimulus, as well as the model parameters.

A.5 Interfacing the ADC test structures on the RAMP chip

During the development of the main neuromorphic system, we also contributed to the development of event-based acquisition of electrophysiological signals by performing the first verification of an event-based ADC designed for this purpose by [Corradi and Indiveri, 2015b]. Here, a system built around the test structure within the RAMP chip was created to record the event based output of the circuit and timestamp it. A reconstruction of the input signal is performed on a PC, based on the logged data. In this project, the programmable system-on-chip which is outlined in the main body of the thesis was used together with the RAMP system (see A.10).

The sensing circuit continuously compares its input to two thresholds θ_{up} and θ_{down}. If the crossing of one of the thresholds is detected, an internal amplified copy of the input signal is set to a reference voltage and an event δ_{up} or δ_{dn} is issued that signifies the polarity of the threshold crossing. After this reset to the reference voltage, the signal is integrated again until the next threshold crossing. Conceptually, in this manner, a delta-encoding of the input is performed. The output of the circuit can be reconstructed through an increment or decre-

ment of a variable by θ_{up} or θ_{down} respectively at the given timestep. This process is outlined in the following.

A.5.1 Signal Reconstruction

In order to perform an initial verification, we provide the test-circuit with an electrophysiological recording which is replayed through a host PC's sound card. The amplitude of this signal is calibrated to be in the μV-range, using an oscilloscope. On the PC, a data logger captures the circuits output event stream. After parsing the log-file, three different series are extracted with

$$\Delta_{dn} = (\delta_{dn_n})_{n \in \{0,..,n_{max}\}}$$

$$\Delta_{up} = (\delta_{up_n})_{n \in \{0,..,n_{max}\}}$$

$$ISI = (ISI_n)_{n \in \{0,..,n_{max}\}}$$

, where δ are the 'up' and 'down' spikes coming from the test circuit. Every event is logged together with a timestamp with $20ns$ resolution. This timestamp is relative to the previously recorded event, in other words, it forms an inter-spike-interval, ISI.

Therefore, the first step in reconstructing the recorded signal is to compute the cumulative sum of ISIs to $T = (t_n)_{n \in \{0,..,n_{max}\}}$, with

$$t_n = \sum_{i=0}^{n} ISI_i$$

Similarly, the reconstruction of the samples associated with the times t_n are computed by summation:

$$s_n = \theta_{up} \cdot \sum_{i=0}^{n} \delta_{up_i} - \theta_{dn} \cdot \sum_{i=0}^{n} \delta_{dn_i};\ S = (s_n)_{n \in \{0,..,n_{max}\}}$$

Here, θ_{up} and θ_{dn} are the thresholds that the pre-amplified signal is compared to. If θ is crossed, a spike is released.

The step from amplitudes s_n at times t_n to a regular sampling is done through interpolation of S to values that are computed for the time-steps

$$X = (x_n)_{x_0=0,\ x_n=x_{n-1}+\frac{1}{r}};\ n \in \{0,..,n_{max}\}$$

, where r is the sample rate of the signal fed into the amplifier via the sound card. In this way, a comparable signal is provided.

The interpolation step itself is linear:

$$y_i = s_{n-1} + (x_i - t_{n-1})\frac{s_n - s_{n-1}}{t_n - t_{n-1}};\ t_{n-1} \leq x_i < t_n$$

The signal Y exhibits low-frequency drift due to the spontaneous activity of the test-pixel due to noise in the input which results in the firing of $\delta_{up/dn}$ events even if there is no input signal presented. A filtering step is used to reduce this artifact in the signal. The same filtering is applied to the input signal before comparison. Choosing the firing threshold allows the user to choose the quality of the reconstructed signal. The choice of small thresholds causes an improvement in the reconstruction but also increases the data throughput in the system. Considering the large number of circuits in a sensor array, bandwidth will become a limiting factor on signal quality. Here, it may be beneficial to set a hard limit on the desired data rate and adaptively tune firing thresholds such that this limit is always

approached, but never crossed. As long as the decoder logs the changes in threshold, the reconstruction would always be of the highest achievable quality.

A.6 An Internet Based Distributed Biohybrid Neuromorphic Setup

In parallel to the algorithm development for structural plasticity, exploratory work on the creation of a physically distributed biohybrid system was performed. The following work is presented in [Serb et al., 2017] and described here in paraphrased form. The distributed biohybrid system relies on the UDP protocol for the communication of events from a cell culture to a neuromorphic system and vice-versa. In this work, the connection weight between source and destination was implemented via memristors at a third, remote location. *UNIPD* denotes the biological setup, located at the University of Padua. *UZH* denotes the neuromorphic side, located at the University of Zurich. *SOTON* denotes the array of memristors that store connection weights at the University of Southampton. Parts of the provided section are written in collaboration with A.Serb, C.Mayr, and T.Podromakis.

A.6.1 Neuromorphic Side

In this experiment, the neuromorphic side of the biohybrid is formed by the neuromorphic system, described in Chapter 3. The FPGA handles the interfacing of the processor with a PC through which events are recorded and transmitted via UDP. In order for the co-processor to be able to process data in real-time, this processor is used 'bare-metal' without

an operating system that might introduce time-varying delays. VHDL-coded peripheral blocks assist and provide interfaces. On the PC side of the USB connection, low-level drivers developed in the C-programming language transmit and receive data byte-wise and make it accessible to a python API that forms the bridge to UDP. Here, data is accumulated and formatted into packets that are sent over the internet to a remote host.

A.6.2 Biological Side

The biological side was initially replaced by a simulation on a dedicated computer to reproducibly test several scenarios of the operation of the overall system. The simulated UNIPD setup received events via UDP and decoded them. As events are received, the setup responds by issuing a response with a given probability. This response is delayed by a predefined offset plus a random delay that affects the timing of the 'post' spike in a spike-timing dependent synaptic plasticity (STDP) rule. In cases in which the delay in response is so large that the response is close to the next incoming 'pre' event, this leads to a depression of the synaptic weight as defined by the plasticity rule. However, if the response event follows temporally close to the received 'pre' event, and within the time-window of the STDP algorithm implemented at SOTON, the synapse is potentiated (the weight increases).

A.6.3 Interfacing of the Neuromorphic and Biological Setups

For the application in a remote closed-loop experiment with memristive plasticity, a protocol was defined which allows iden-

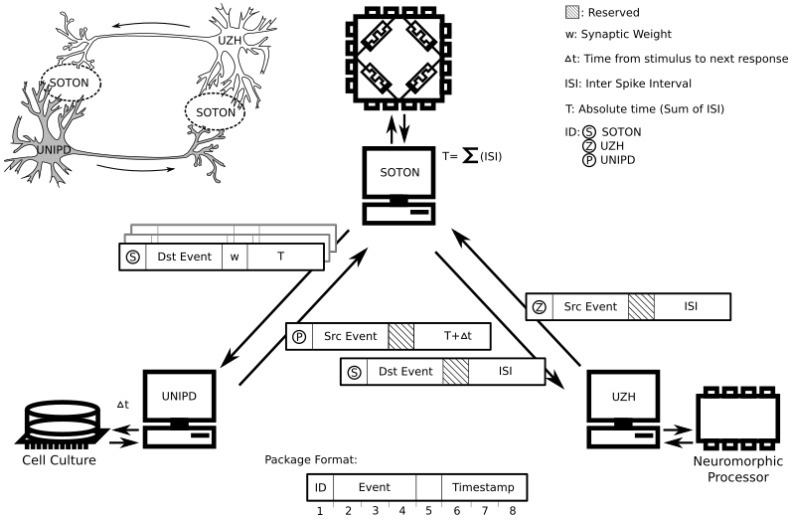


Figure A.11:
Schematic illustration of the network protocol for closed-loop experiments via the extended UDP specification

tification of the senders on the receiver side. This allows interpretation of an arriving packet independent of the possibly changing IP addresses of the senders and instructs the receiver how to process the incoming data. The communication scheme developed follows the protocol illustrated in Figure A.11.

The signal chain begins with the UZH setup, which consists of a PC that interfaces to the ROLLS chip via a USB interface. Events coming from the neuromorphic processor are time-stamped with inter-spike intervals (ISI) and transmitted to the SOTON setup along with the identity of the source neuron. A unique identifier in the first octet of the payload allows the SOTON setup to recognize the $24bit$ timestamp as ISI information.

At SOTON the system receives the events, translates the

ISI timestamps into absolute time (T) and consults an adjacency matrix in list format in order to determine which memristor synapses need to treat the incoming spike as a pre-synaptic event. The affected synapses then register the spike as a pre-synaptic event, their resistive states are assessed by the memristor-driving setup (ArC), and results are mapped to a synaptic weight. Subsequently, SOTON sends a series of address-events to UNIPD which contain: (a) the SOTON identifier, (b) the ID of the target synapse to stimulate, (c) a measure of the weight (8 bits) and (d) the timestamp of the 'Post-Synaptic Potential' (PSP) in absolute time T. UNIPD can then receive this information, and deliver stimulation where required. Importantly, UNIPD uses the weight to determine the intensity of stimulation on each target and the timestamp in order to set its own time reference. Any neuron that fires at the UNIPD setup will be timestamped in absolute time as determined by the amount of 'physical time' ($T + \delta t$) elapsed between the last synaptic PSP arriving at the said neuron and the time of firing. For example, if neuron N emits a spike twelve time steps after the last PSP (arriving at for example at an absolute time value (AT) of 15000), UNIPD will send an AER packet to SOTON informing the system that cell N has fired at AT 15012. The UNIPD response to SOTON thus allows to relate the firing of each post-synaptic neuron to an absolute time frame and compute when and where plasticity should be implemented. In this way, the protocol described enables compensation for transmission-imposed delays in the time-critical aspects of the system where STDP is performed. The approach of uniquely identifying the senders in a reserved octet is a method that can be extended for remote configuration and other uses in the future through using the involved bits to encode instructions.

In order to verify the remote setup described above, a work-

station at SOTON receives events as UDP packets from both the 'presynaptic' neuromorphic circuits at UZH and 'postsynaptic' events from UNIPD (currently in the form of a simulated cell culture). According to the timing of both types of events, a linear spike-timing dependent plasticity rule was implemented that determines the increase or decrease of the synaptic weight between the pre- and postsynaptic site.

To implement this plasticity algorithm at SOTON, an array of memristors was used to update and store the weights, indicating the synaptic efficacies. Figure A.12 shows the course of the memristors resistive state over the number of weight updates on the x-axis. Since the simulated postsynaptic site had a higher probability of firing in correlation with incoming events coming from UZH (routed through SOTON) we see a general increase in the stored weight with only a little depression due to anticorrelated timing (see Figure A.12).

After this verification of the communication infrastructure, we added an actual biological system into the loop. Ath the time of writing, this work has been submitted to *Nature Nanotechnology*. The following data and descriptions have been created in collaboration with *Alexantrou Serb, Andrea Corna, Ali Khiat, Federico Rocchi, Marco Reato, Marta Maschietto, Christian Mayr, Giacomo Indiveri, Stefano Vassanelli and Themistoklis Prodromakis*.

In this proof of principle, a simple three neuron network is connected in a feed-forward way whereby a presynaptic artificial neuron (ANPRE) connects to a biological neuron (BN), which in turn connects to another, post-synaptic, artificial neuron (ANPOST). (Figure A.13)

All connections are realized via memristive synapses. Notably, the artificial and biological neuron setups communicate exclusively via the synapse setup. The central position of the synapse setup within the network renders it the control center

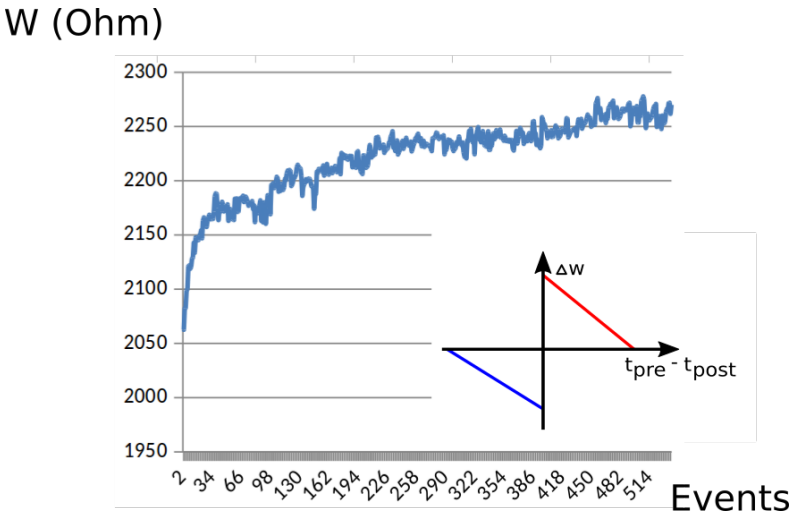


Figure A.12:
Weight evolution as measured from the memristor Array as verification of the implemented UDP Protocol. The resistive state of the memristor that holds the Weight of the connection between a neuromorphic and a biological neuron in Ohm is potentiated over time as pairs of source and destination events lead to weight updates through a linear synaptic plasticity rule (illustration, bottom right)

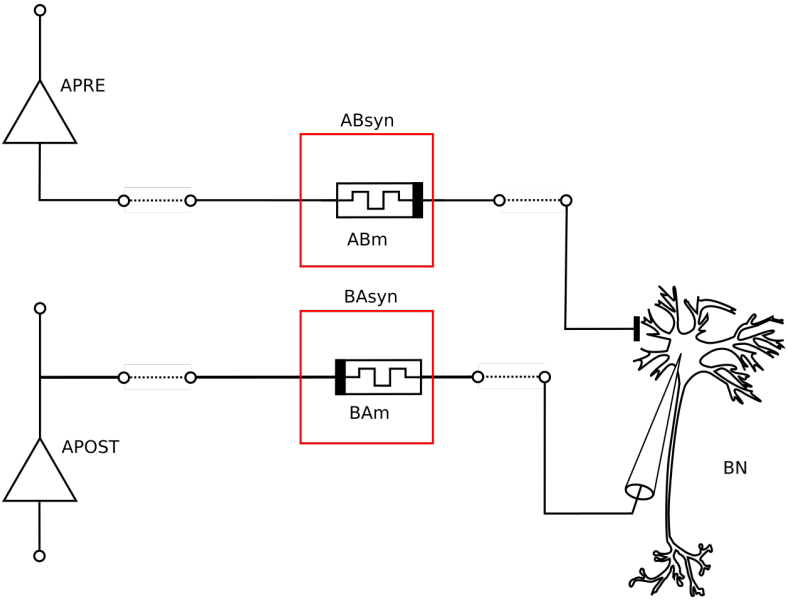


Figure A.13:
Geographically distributed bNN setup. High-level connectivity diagram. bNN layout, with two memristors (BAm and ABm) establishing the two synaptors: ABsyn (Forward path with BN presynaptic to APOST) and BAsyn (Backward path, with APRE presynaptic to BN). ABsyn: APRE spikes are fed to ABm whose resistivity changes tune extracellular stimulation of BN through a capacitive microelectrode array. BAsyn: BN activity (sub- and supra-threshold) is monitored by a patch-clamp pipette. Spikes are fed to BAm which drives APOST firing by modulating the EPSC

of the entire system. For that reason, it holds both the connectivity matrix defining the network connectivity and the record of all spiking activity in addition to controlling the handling of time. The plasticity rule used for the reported benchmarking experiment is a modified, rate-coded Bienenstock-Cooper-Munro (BCM) model, where plasticity (Long Term Potentiation (LTP) and Depression (LTD) occurs upon postsynaptic spiking with the direction of plasticity (i.e., potentiation, depression or no plasticity) determined by the frequency of pre-synaptic activity within a time window leading to post-synaptic spikes. In our modified version, this strictly applies to the reverse biological-to-artificial pathway. In fact, in the forward pathway, plasticity is initiated only by high-frequency pre-synaptic activity, i.e., emulating a short-term post-tetanic potentiation (PTP) mechanism, eventually leading to LTP of the synapse. It should be noted that using a rate-coded, and not phase-coded, plasticity rule provides a certain degree of immunity against physical and location-dependent network delays in this experiment, as the specific timing of spikes is secondary in importance to the overall rate.

Results from the backward pathway are shown in Figure A.15. The spiking activity of BN is forwarded, through memristor BAm, to its post-synaptic target, in this case, the artificial neuron ANPOST. ANPOST is set to be spontaneously active in order to guarantee that BCM plasticity will be triggered sufficiently often during the experiment. The effect of spiking at BN then modulates this spontaneous activity. This is evidenced by the increased activity observed in ANPOST coinciding with spiking at BN. This activity dies away as soon as LTD takes place on the forward path, as demonstrated in Figure A.15b. The corresponding memristive synaptic weight evolution is shown in Figure A.15c and reveals the underlying plasticity. Notably, in contrast to the forward path where

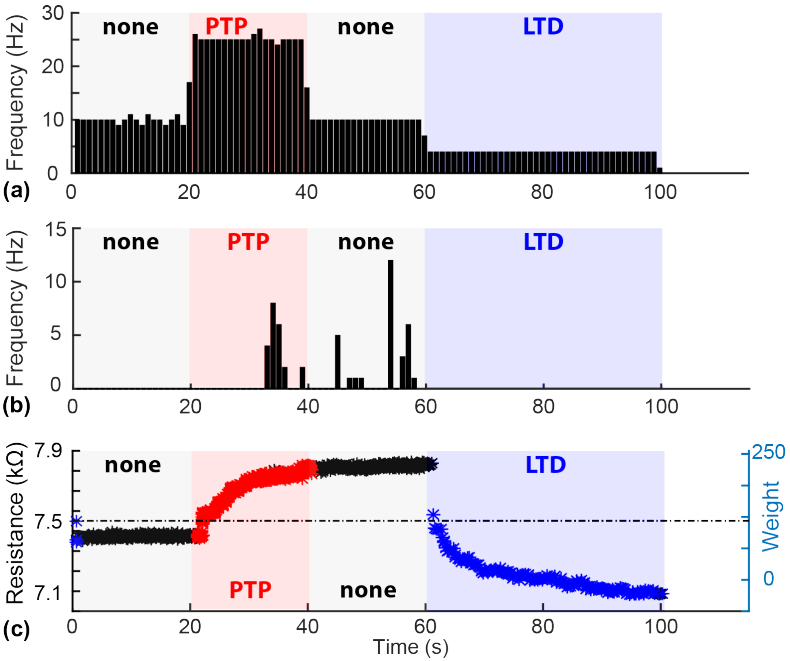


Figure A.14:
Geographically distributed bNN experimentally observed operation: Forward path. (a) Activity pattern of artificial neuron ANPRE. Firing frequency is modulated in four phases designed to induce plasticity in the synapse linking to the biological neuron (BN) in the direction indicated by the sequence: none/LTP/none/LTD, as per the modified BCM plasticity model. (b) Spikes response of biological neuron BN to stimulation from ANPRE. Upon induction, in the late stages of LTP potentiation, BN begins responding by firing action potentials (APs). These persist until the commencement of the LTD depression phase. (c) Synaptic weight evolution at the forward (ANPRE to BN) pathway. Each data point denotes a synaptic event that caused LTP (red), LTD (blue) or no further plasticity changes (black)

tight control of stimulation at ANPRE to BN synapse guarantees reliable induction of LTP and LTD, the persistently low activity at BN leads to either LTD or no plasticity at the BN to ANPOST synapse.

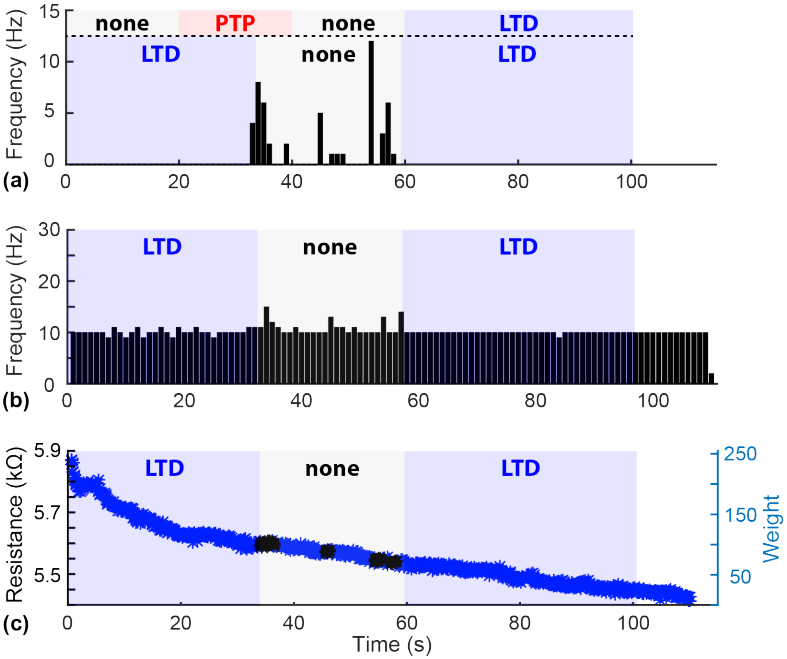


Figure A.15:
Geographically distributed bNN experimentally observed operation: Backward path. (a) Firing response of biological neuron BN to stimulation from ANPRE as reported in Figure A.14a. Shadowed areas indicate plasticity changes occurring at the ABsyn (above the dashed line) and BAsyn (below the dashed line). As previously noted, BN begins responding by firing action potentials during the late stages of potentiation. These persist until the commencement of the LTD induction phase. (b) Spiking frequency measured at artificial neuron ANPOST; the post-synaptic target of BN. Increase in spiking activity is observed in the middle of the run in coincidence in response to more frequent stimulation from BN followed by a return to baseline, spontaneous activity towards the end. (c) Synaptic weight evolution at the reverse (BN to ANPOST) pathway. Each data point denotes a synaptic event that induced LTP (red), LTD (blue) or no plasticity (black). The rather infrequent activity observed at BN caused the spike rate-dependent plasticity (SRDP) plasticity BCM rule to lead to strong LTDP at the reverse pathway synapse and create the LTD/none/LTD plasticity induction pattern shown via blue / gray shadings throughout all panels. LTP;LTD;none plasticity induction window frames used in Figure A.14 are reproduced at the top of A.15a for reference. X-axis common to all panels.

A.7 Developmental Cell Culture Data

The following data was acquired by *M. Maschietto* and used to develop an approach for modeling developmental processes involved in network formation in *in vitro* cultures.

A.7.1 Synapsin1 labelling

Data on the formation of functional synapses (see Figure A.16) over time was acquired through the following procedure:

- E18/E19 rat hippocampal neurons were labeled by indirect immunofluorescence with an antibody specific for pre-synaptic Synapsin 1
- Images were acquired at 63X magnification at video-confocal microscope, with z-stack procedure, from neurons at different development stages (from DIV 6 to DIV 30)
- Merges of the z-stacks were created in order to collect the dots from all the confocal planes
- For each neuron, the length of each branch was measured in μm and the number of Synapsin 1 do. were counted
- The number of dots per μm length was calculated for each branch
- The mean number of dots per μm length and SEM were calculated for each DIV (considering as "n" the total number of branches analyzed)

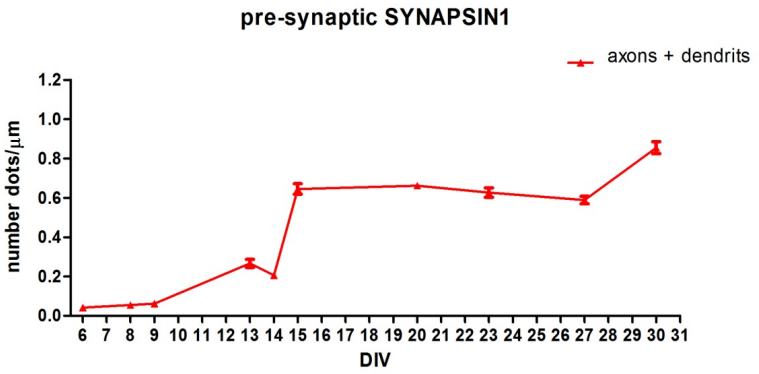


Figure A.16:
Indirect immunofluorescence labeling with an antibody specific for pre-synaptic Synapsin 1 allows to estimate the density of functional synapses over time (DIV)

A.7.2 Sholl Analysis

For the rough estimation of synapse density, Sholl analysis gives an indication of how the neuron extends and where ramifications are observable, as a function of the distance to the cells soma (see Figure A.17). Note that synapse density per area of cell membrane was assumed to be distant independent, an assumption which only allows a rough estimate of the density of potential connections.

Sholl analysis creates a series of concentric shells around the focus of a neuronal arbor, and counts how many times the arbor intersects the sampling shells A starting radius (the radius of the smallest sampling circle, i.e., the first distance to be sampled), an ending radius (the radius of the largest (last) sampling circle) and the radius step size (the sampling interval between radii of consecutive sampling circles) are fixed prior

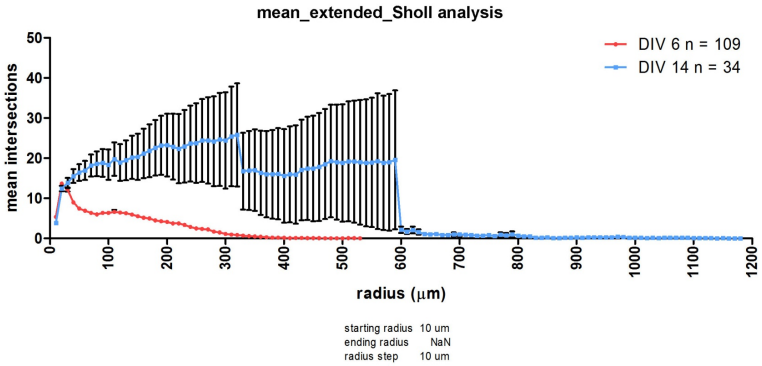


Figure A.17:
Sholl analysis for E18/E19 rat hippocampal neurons in vitro. Data was accumulated for n neurons at DIV6 and DIV14

to the analysis. For each transfected neuron, that must be isolated (that is, among non-transfected neurons), the analysis provides with the number of intersections for each circle.

A.7.3 Maximum Linear Distance of Neuronal Processes

The procedure for acquiring measurements of how far neurites extend from the soma (see Figure A.18) was performed as follows:

- Neurons were transfected with cytoplasmic EGFP protein to better visualize the branching
- Images were analyzed at DIV 6 a. DIV 14 (the transfection efficiency decreases with increasing DIV, so it is not possible to have data at more than DIV 14)

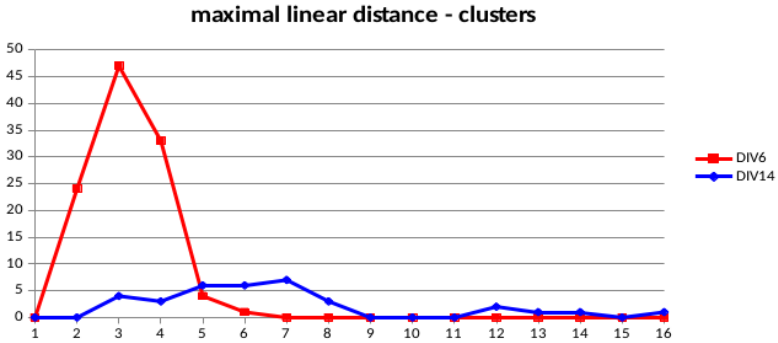


Figure A.18:
Maximum linear distance from soma to the end of a neurite, measured over time

- Only isolated transfected neurons were analyzed (that is, one neuron transfected among many others not transfected)
- For each transfected neuron, the maximal linear length of branching was measured, starting from the beginning of the branch on the soma
- The maximal linear lengths were grouped in clusters of 100 i.tm length, in order to see if there is a preferential clustering of the neurons
- The clustered data were plotted also normalized on the total number of neurons

This procedure gives rise to the following data, which was used to determine the maximal size of the rotation-symmetric synapse probability kernel.

References

L. F. Abbott, S. Song, and K. D. Miller. Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, 3(9):919–926, sep 2000. doi: 10.1038/78829. URL *https://doi.org/10.1038/78829*.

S. A. Bamford, A. F. Murray, and D. J. Willshaw. Synaptic rewiring for topographic mapping and receptive field development. *Neural Networks*, 23(4):517–527, may 2010. doi: 10.1016/j.neunet.2010.01.005. URL *https://doi.org/10.1016/j.neunet.2010.01.005*.

C. Bartolozzi and G. Indiveri. Synaptic dynamics in analog VLSI. *Neural Computation*, 19(10):2581–2603, Oct 2007. doi: 10.1162/neco.2007.19.10.2581. URL *http://ncs.ethz.ch/pubs/pdf/Bartolozzi_Indiveri07.pdf*.

C. Bats, L. Groc, and D. Choquet. The interaction between stargazin and PSD-95 regulates AMPA receptor surface trafficking. *Neuron*, 53(5):719–734, mar 2007. doi: 10.1016/j.neuron.2007.01.030. URL *https://doi.org/10.1016/j.neuron.2007.01.030*.

G.-Q. Bi and M.-M. Poo. Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, 18(24):10464–10472, 1998.

F. Boi, T. Moraitis, V. D. Feo, F. Diotalevi, C. Bartolozzi, G. Indiveri, and A. Vato. A bidirectional brain-machine in-

terface featuring a neuromorphic hardware decoder. *Frontiers in Neuroscience*, 10, dec 2016. doi: 10.3389/fnins.2016.00563. URL *https://doi.org/10.3389/fnins.2016.00563*.

T. Bonhoeffer and F. Engert. Dendritic spine changes associated with hippocampal long-term synaptic plasticity. *Nature*, 399(6731):66–70, may 1999. doi: 10.1038/19978. URL *https://doi.org/10.1038/19978*.

J. Brader, W. Senn, and S. Fusi. Learning real world stimuli in a neural network with spike-driven synaptic dynamics. *Neural Computation*, 19:2881–2912, 2007.

R. Brette and W. Gerstner. Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of Neurophysiology*, 94:3637–3642, 2005.

M. Butz and A. van Ooyen. A simple rule for dendritic spine and axonal bouton formation can account for cortical reorganization after focal retinal lesions. *PLoS Computational Biology*, 9(10):e1003259, oct 2013. doi: 10.1371/journal.pcbi.1003259. URL *https://doi.org/10.1371/journal.pcbi.1003259*.

M. Butz, F. Wörgötter, and A. van Ooyen. Activity-dependent structural plasticity. *Brain Research Reviews*, 60(2):287–305, may 2009. doi: 10.1016/j.brainresrev.2008.12.023. URL *https://doi.org/10.1016/j.brainresrev.2008.12.023*.

W. Cerpa, A. Gambrill, N. C. Inestrosa, and A. Barria. Regulation of NMDA-receptor synaptic transmission by wnt signaling. *Journal of Neuroscience*, 31(26):9466–9471, jun 2011. doi: 10.1523/jneurosci.6311-10.2011. URL *https://doi.org/10.1523/jneurosci.6311-10.2011*.

E. Chicca, F. Stefanini, C. Bartolozzi, and G. Indiveri. Neuromorphic electronic circuits for building autonomous cognitive systems. *Proceedings of the IEEE*, 102(9):1367–1388, 9 2014. ISSN 0018-9219. doi: 10.1109/JPROC.2014.2313954. URL *http://ncs.ethz.ch/pubs/pdf/Chicca_etal14.pdf*.

M. Chistiakova, N. M. Bannon, J.-Y. Chen, M. Bazhenov, and M. Volgushev. Homeostatic role of heterosynaptic plasticity: models and experiments. *Frontiers in Computational Neuroscience*, 9, jul 2015. doi: 10.3389/fncom.2015.00089. URL *https://doi.org/10.3389/fncom.2015.00089*.

W.-S. Chung, N. J. Allen, and C. Eroglu. Astrocytes control synapse formation, function, and elimination. *Cold Spring Harbor Perspectives in Biology*, 7(9):a020370, feb 2015. doi: 10.1101/cshperspect.a020370. URL *https://doi.org/10.1101/cshperspect.a020370*.

F. Corradi and G. Indiveri. A neuromorphic event-based neural recording system for smart brain-machine-interfaces. *IEEE Transactions on Biomedical Circuits and Systems*, 9 (5):699–709, oct 2015a. doi: 10.1109/tbcas.2015.2479256. URL *https://doi.org/10.1109/tbcas.2015.2479256*.

F. Corradi and G. Indiveri. A neuromorphic event-based neural recording system for smart brain-machine-interfaces. *Biomedical Circuits and Systems, IEEE Transactions on*, 9(5):699–709, 2015b. doi: 10.1109/TBCAS.2015.2479256. URL *http://ncs.ethz.ch/pubs/pdf/Corradi_Indiveri15.pdf*.

M. Deger, A. Seeholzer, and W. Gerstner. Multi-contact synapses for stable networks: a spike-timing dependent model of dendritic spine plasticity and turnover, 2016.

S. Deiss, T. Delbruck, R. Douglas, M. Fischer, M. Mahowald, T. Matthews, and A. Whatley. Address-event asynchronous

local broadcast protocol. World Wide Web page, 1994. http://www.ini.uzh.ch/~amw/scx/aeprotocol.html.

T. Delbruck, R. Berner, P. Lichtsteiner, and C. Dualibe. 32-bit configurable bias current generator with sub-off-current capability. In *International Symposium on Circuits and Systems, (ISCAS), 2010*, pages 1647–1650, Paris, France, 2010. IEEE, IEEE. doi: 10.1109/ISCAS.2010.5537475.

J. Dragas, V. Viswam, A. Shadmani, Y. Chen, R. Bounik, A. Stettler, M. Radivojevic, S. Geissler, M. E. J. Obien, J. Müller, and A. Hierlemann. In vitro multi-functional microelectrode array featuring 59 760 electrodes, 2048 electrophysiology channels, stimulation, impedance measurement, and neurotransmitter detection channels. *IEEE Journal of Solid-State Circuits*, 52(6):1576–1590, June 2017. ISSN 0018-9200. doi: 10.1109/JSSC.2017.2686580.

D. B. Fasnacht. *Experimentation Platforms for Neuromorphic Event-Based Multi-Chip Systems*. Doctoral Thesis, ETH Zurich, 2016. URL *https://www.research-collection.ethz.ch/handle/20.500.11850/123065?show=full*. 00000 DOI: 10.3929/ethz-a-010785412.

M. Fauth, F. Wörgötter, and C. Tetzlaff. The formation of multi-synaptic connections by the interaction of synaptic and structural plasticity and their functional consequences. *PLOS Computational Biology*, 11(1):e1004031, jan 2015. doi: 10.1371/journal.pcbi.1004031. URL *https://doi.org/10.1371/journal.pcbi.1004031*.

S. Feldt, P. Bonifazi, and R. Cossart. Dissecting functional connectivity of neuronal microcircuits: experimental and theoretical insights. *Trends in Neurosciences*, 34(5):225–

236, may 2011. doi: 10.1016/j.tins.2011.02.007. URL *https://doi.org/10.1016/j.tins.2011.02.007*.

K. Futai, M. J. Kim, T. Hashikawa, P. Scheiffele, M. Sheng, and Y. Hayashi. Retrograde modulation of presynaptic release probability through signaling mediated by PSD-95–neuroligin. *Nature Neuroscience*, 10(2):186–195, jan 2007. doi: 10.1038/nn1837. URL *https://doi.org/10.1038/nn1837*.

F. Galluppi, C. Denk, M. C. Meiner, T. C. Stewart, L. A. Plana, C. Eliasmith, S. Furber, and J. Conradt. Event-based neural computing on an autonomous mobile platform. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2014. doi: 10.1109/icra.2014.6907270. URL *https://doi.org/10.1109/icra.2014.6907270*.

R. George and G. Indiveri. Tunable device-mismatch effects for stochastic computation in analog/digital neuromorphic computing architectures. In *2016 IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pages 77–80, Dec 2016. doi: 10.1109/ICECS.2016.7841136.

R. George, C. Mayr, G. Indiveri, and S. Vassanelli. Event-based softcore processor in a biohybrid setup applied to structural plasticity. In *Event-based Control, Communication, and Signal Processing (EBCCSP), 2015 International Conference on*, pages 1–4, June 2015a. doi: 10.1109/EBCCSP.2015.7300664. URL *http://ncs.ethz.ch/pubs/pdf/George_etal15.pdf*.

R. George, E. Covi, J. Frascaroli, S. Brivio, H. Mostafa, C. Mayr, S. Spiga, and G. Indiveri. Spike-based learning

in a hybrid cmos/ hfo2 memristive neural network (in review). *JETCAS*, 2017.

R. M. George, P. U. Diehl, M. Cook, C. Mayr, and G. Indiveri. Modeling the interplay between structural plasticity and spike-timing-dependent plasticity. *BMC Neuroscience*, 16(1):P107, Dec 2015b. ISSN 1471-2202. doi: 10.1186/1471-2202-16-S1-P107. URL *https://doi.org/10.1186/1471-2202-16-S1-P107*.

W. Gong, J. Senčar, D. J. Bakkum, D. Jäckel, M. E. J. Obien, M. Radivojevic, and A. R. Hierlemann. Multiple single-unit long-term tracking on organotypic hippocampal slices using high-density microelectrode arrays. *Frontiers in Neuroscience*, 10, nov 2016. doi: 10.3389/fnins.2016.00537. URL *https://doi.org/10.3389/fnins.2016.00537*.

D. F. M. Goodman. The brian simulator. *Frontiers in Neuroscience*, 3(2):192–197, sep 2009. doi: 10.3389/neuro.01.026.2009. URL *https://doi.org/10.3389/neuro.01.026.2009*.

A. Govindarajan, I. Israely, S.-Y. Huang, and S. Tonegawa. The dendritic branch is the preferred integrative unit for protein synthesis-dependent LTP. *Neuron*, 69(1):132–146, jan 2011. doi: 10.1016/j.neuron.2010.12.008. URL *https://doi.org/10.1016/j.neuron.2010.12.008*.

S. L. Graham, P. B. Kessler, and M. K. Mckusick. Gprof: A call graph execution profiler. *SIGPLAN Not.*, 17(6):120–126, June 1982. ISSN 0362-1340. doi: 10.1145/872726.806987. URL *http://doi.acm.org/10.1145/872726.806987*.

I. Gupta, A. Serb, A. Khiat, R. Zeitler, S. Vassanelli, and T. Prodromakis. Real-time encoding and compression of

neuronal spikes by metal-oxide memristors. *Nature Communications*, 7:12805, sep 2016. doi: 10.1038/ncomms12805. URL *https://doi.org/10.1038/ncomms12805*.

S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *ArXiv e-prints*, 2015.

D. Hebb. *The organization of behavior*. Wiley: New York, 1949.

A. Hodgkin and A. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, 117:500–44, 1952.

A. Holtmaat and K. Svoboda. Experience-dependent structural synaptic plasticity in the mammalian brain. *Nature Reviews Neuroscience*, 10(9):647–658, sep 2009. doi: 10.1038/nrn2699. URL *https://doi.org/10.1038/nrn2699*.

S. Hussain, R. Gopalakrishnan, A. Basu, and S.-C. Liu. Morphological learning: Increased memory capacity of neuromorphic systems with binary synapses exploiting AER based reconfiguration. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*. IEEE, aug 2013. doi: 10.1109/ijcnn.2013.6706928. URL *https://doi.org/10.1109/ijcnn.2013.6706928*.

M. Hutzler and P. Fromherz. Silicon chip with capacitors and transistors for interfacing organotypic brain slice of rat hippocampus. *European Journal of Neuroscience*, 19(8):2231–2238, apr 2004. doi: 10.1111/j.0953-816x.2004.03311.x. URL *https://doi.org/10.1111/j.0953-816x.2004.03311.x*.

G. Indiveri, F. Corradi, and N. Qiao. Neuromorphic architectures for spiking deep neural networks. In *Electron Devices*

Meeting (IEDM), 2015 IEEE International, pages 4.2.1–4.2.14. IEEE, Dec. 2015. URL *http://ncs.ethz.ch/pubs/pdf/Indiveri_etal15.pdf*.

G. Indiveri, F. Corradi, and N. Qiao. Neuromorphic architectures for spiking deep neural networks, 2016. URL *http://rgdoi.net/10.13140/RG.2.1.2082.1527*.

S. P. Jenkinson, D. Grandgirard, M. Heidemann, A. Tscherter, M.-A. Avondet, and S. L. Leib. Embryonic stem cell-derived neurons grown on multi-electrode arrays as a novel in vitro bioassay for the detection of clostridium botulinum neurotoxins. *Frontiers in Pharmacology*, 8, feb 2017. doi: 10.3389/fphar.2017.00073. URL *https://doi.org/10.3389/fphar.2017.00073*.

P. Jourdain, K. Fukunaga, and D. Muller. Calcium/calmodulin-dependent protein kinase ii contributes to activity-dependent filopodia growth and spine formation. *Journal of Neuroscience*, 23 (33):10645–10649, 2003. ISSN 0270-6474. URL *http://www.jneurosci.org/content/23/33/10645*.

D. Kappel, S. Habenschuss, R. Legenstein, and W. Maass. Network plasticity as bayesian inference. *PLOS Computational Biology*, 11(11):e1004485, nov 2015. doi: 10.1371/journal.pcbi.1004485. URL *https://doi.org/10.1371/journal.pcbi.1004485*.

T. Kleindienst, J. Winnubst, C. Roth-Alpermann, T. Bonhoeffer, and C. Lohmann. Activity-dependent clustering of functional synaptic inputs on developing hippocampal dendrites. *Neuron*, 72(6):1012–1024, dec 2011. doi: 10.1016/j.neuron.2011.10.015. URL *https://doi.org/10.1016/j.neuron.2011.10.015*.

A. Knoblauch, E. Körner, U. Körner, and F. T. Sommer. Structural synaptic plasticity has high memory capacity and can explain graded amnesia, catastrophic forgetting, and the spacing effect. *PLoS ONE*, 9(5):e96485, may 2014. doi: 10.1371/journal.pone.0096485. URL *https://doi.org/10.1371/journal.pone.0096485*.

D. E. Knuth. Dancing links. *arXiv*, 2000.

A. Lambacher, V. Vitzthum, R. Zeitler, M. Eickenscheidt, B. Eversmann, R. Thewes, and P. Fromherz. Identifying firing mammalian neurons in networks with high-resolution multi-transistor array (mta). *Applied Physics A*, 102(1):1–11, Jan 2011. ISSN 1432-0630. doi: 10.1007/s00339-010-6046-9. URL *https://doi.org/10.1007/s00339-010-6046-9*.

R. Lamprecht and J. LeDoux. Structural plasticity and memory. *Nature Reviews Neuroscience*, 5(1):45–54, jan 2004. doi: 10.1038/nrn1301. URL *https://doi.org/10.1038/nrn1301*.

P. Lichtsteiner, C. Posch, and T. Delbruck. A 128x128 120 dB 30 mW asynchronous vision sensor that responds to relative intensity change. In *2006 IEEE ISSCC Digest of Technical Papers*, pages 508–509. IEEE, Feb 2006.

G. Liu. Local structural balance and functional interaction of excitatory and inhibitory synapses in hippocampal dendrites. *Nature Neuroscience*, 7(4):373–379, mar 2004. doi: 10.1038/nn1206. URL *https://doi.org/10.1038/nn1206*.

S. Liu, A. van Schaik, B. Minch, and T. Delbruck. Event-based 64-channel binaural silicon cochlea with q enhancement mechanisms. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 2027–2030. IEEE, 2010.

P. Livi and G. Indiveri. A current-mode conductance-based silicon neuron for address-event neuromorphic systems. In *International Symposium on Circuits and Systems, (ISCAS), 2009*, pages 2898–2901. IEEE, May 2009. doi: 10.1109/ISCAS.2009.5118408. URL *http://ncs.ethz.ch/pubs/pdf/Livi_Indiveri09.pdf*.

M. London and M. Häusser. DENDRITIC COMPUTATION. *Annual Review of Neuroscience*, 28(1):503–532, jul 2005. doi: 10.1146/annurev.neuro.28.061604.135703. URL *https://doi.org/10.1146/annurev.neuro.28.061604.135703*.

E. Ltd. Raggedstone2 manual. *Issue 1.02 29.06.2010 (Online)*, 2010.

E. Marder. Variability, compensation, and modulation in neurons and circuits. *Proceedings of the National Academy of Sciences*, 108(Supplement_3):15542–15548, mar 2011. doi: 10.1073/pnas.1010674108. URL *https://doi.org/10.1073/pnas.1010674108*.

P. Merolla, J. Arthur, R. Alvarez, J.-M. Bussat, and K. Boahen. A multicast tree router for multichip neuromorphic systems. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 61(3):820–833, March 2014a. ISSN 1549-8328. doi: 10.1109/TCSI.2013.2284184.

P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, Aug 2014b. ISSN

0036-8075, 1095-9203. doi: 10.1126/science.1254642. URL *http://www.sciencemag.org/content/345/6197/668*.

D. Meyer, T. Bonhoeffer, and V. Scheuss. Balance and stability of synaptic structures during synaptic plasticity. *Neuron*, 82(2):430–443, apr 2014. doi: 10.1016/j.neuron.2014.02.031. URL *https://doi.org/10.1016/j.neuron.2014.02.031*.

M. B. Milde, H. Blum, A. Dietmüller, D. Sumislawska, J. Conradt, G. Indiveri, and Y. Sandamirskaya. Obstacle avoidance and target acquisition for robot navigation using a mixed signal analog/digital neuromorphic processing system. *Frontiers in Neurorobotics*, 11, jul 2017. doi: 10.3389/fnbot.2017.00028. URL *https://doi.org/10.3389/fnbot.2017.00028*.

H. Mostafa, A. Khiat, A. Serb, C. G. Mayr, G. Indiveri, and T. Prodromakis. Implementation of a spike-based perceptron learning rule using TiO2-x memristors. *Frontiers in Neuroscience*, 9(357), 2015. doi: 10.3389/fnins.2015.00357.

H. Mostafa, C. Mayr, and G. Indiveri. Beyond spike-timing dependent plasticity in memristor crossbar arrays. In *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, pages 926–929. IEEE, 2016.

M. Noack, J. Partzsch, C. Mayr, S. Hänzsche, S. Scholze, S. Höppner, G. Ellguth, and R. Schüffny. Switched-capacitor realization of presynaptic short-term-plasticity and stop-learning synapses in 28 nm CMOS. *Frontiers in Neuroscience*, 9(10), 2015.

M. Osswald, S.-H. Ieng, R. Benosman, and G. Indiveri. A spiking neural network model of 3d perception for event-based neuromorphic stereo vision systems. *Scientific Reports*, 7:40703, jan 2017. doi: 10.1038/srep40703. URL *https://doi.org/10.1038/srep40703*.

A. Pavasović, A. Andreou, and C. Westgate. Characterization of subthreshold MOS mismatch in transistors for VLSI systems. *Journal of VLSI Signal Processing*, 8(1):75–85, July 1994.

J. M. S. Pearce. The Neuroanatomy of Herophilus. *European Neurology*, 69(5):292–295, 2013. ISSN 0014-3022, 1421-9913. doi: 10.1159/000346232. URL *http://www.karger.com/Article/FullText/346232*. 00011.

N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska, and G. Indiveri. A re-configurable online learning spiking neuromorphic processor comprising 256 neurons and 128k synapses. *Frontiers in Neuroscience*, 9 (141), 2015. ISSN 1662-453X. doi: 10.3389/fnins.2015.00141. URL *http://www.frontiersin.org/neuromorphic_engineering/10.3389/fnins.2015.00141/abstract*.

R. L. Redondo and R. G. M. Morris. Making memories last: the synaptic tagging and capture hypothesis. *Nature Reviews Neuroscience*, 12(1):17–30, jan 2011. doi: 10.1038/nrn2963. URL *https://doi.org/10.1038/nrn2963*.

C. Rossant, D. Goodman, J. Platkiewicz, and R. Brette. Automatic fitting of spiking neuron models to electrophysiological recordings. *Frontiers in Neuroinformatics*, pages 1–14, 2010. ISSN ISSN 1662-5196. doi: 10.3389/neuro.11/002.2010.

S. B. Rosso and N. C. Inestrosa. WNT signaling in neuronal maturation and synaptogenesis. *Frontiers in Cellular Neuroscience*, 7, 2013. doi: 10.3389/fncel.2013.00103. URL *https://doi.org/10.3389/fncel.2013.00103*.

S. Roy and A. Basu. An Online Structural Plasticity Rule for Generating Better Reservoirs. *Neural Computation*, 28

(11):2557–2584, Sept. 2016. ISSN 0899-7667. doi: 10.1162/NECO_a_00886. URL *https://doi.org/10.1162/NECO_a_00886*. 00000.

S. Royer and D. Paré. Conservation of total synaptic weight through balanced synaptic depression and potentiation. *Nature*, 422(6931):518–522, apr 2003. doi: 10.1038/nature01530. URL *https://doi.org/10.1038/nature01530*.

A. Serb, A. Corna, R. George, A. Khiat, F. Rocchi, M. Reato, M. Maschietto, C. Mayr, G. Indiveri, S. Vassanelli, and T. Prodromakis. A geographically distributed bio-hybrid neural network with memristive plasticity. *arXiv*, 2017.

T. Serrano-Gotarredona and B. Linares-Barranco. Systematic width-and-length dependent CMOS transistor mismatch characterization and simulation. *Analog Integrated Circuits and Signal Processing*, 21(3):271–296, December 1999.

A. A. Sharp, M. B. O'Neil, L. F. Abbott, and E. Marder. Dynamic clamp: computer-generated conductances in real neurons. *Journal of Neurophysiology*, 69(3):992–995, 1993. ISSN 0022-3077. URL *http://jn.physiology.org/content/69/3/992*.

M. A. Smith, G. C. R. Ellis-Davies, and J. C. Magee. Mechanism of the distance-dependent scaling of schaffer collateral synapses in rat CA1 pyramidal neurons. *The Journal of Physiology*, 548(1):245–258, feb 2003. doi: 10.1113/jphysiol.2002.036376. URL *https://doi.org/10.1113/jphysiol.2002.036376*.

R. Spiess, R. George, M. Cook, and P. U. Diehl. Structural plasticity denoises responses and improves learning speed. *Frontiers in Computational Neuroscience*, 10, sep 2016. doi:

10.3389/fncom.2016.00093. URL *https://doi.org/10.3389/fncom.2016.00093*.

K. Staras, T. Branco, J. J. Burden, K. Pozo, K. Darcy, V. Marra, A. Ratnayaka, and Y. Goda. A vesicle superpool spans multiple presynaptic terminals in hippocampal neurons. *Neuron*, 66(1):37–44, apr 2010. doi: 10.1016/j.neuron.2010.03.020. URL *https://doi.org/10.1016/j.neuron.2010.03.020*.

J. Sulzer, S. Haller, F. Scharnowski, N. Weiskopf, N. Birbaumer, M. Blefari, A. Bruehl, L. Cohen, R. deCharms, R. Gassert, R. Goebel, U. Herwig, S. LaConte, D. Linden, A. Luft, E. Seifritz, and R. Sitaram. Real-time fMRI neurofeedback: Progress and challenges. *NeuroImage*, 76:386–399, aug 2013. doi: 10.1016/j.neuroimage.2013.03.033. URL *https://doi.org/10.1016/j.neuroimage.2013.03.033*.

S. Vassanelli. Multielectrode and multitransistor arrays for in vivo recording. In *Nanotechnology and Neuroscience: Nano-electronic, Photonic and Mechanical Neuronal Interfacing*, pages 239–267. Springer New York, 2014. doi: 10.1007/978-1-4899-8038-0_8. URL *https://doi.org/10.1007/978-1-4899-8038-0_8*.

J. Weidendorfer. kcachegrind.github.io. *Copyright*, 2003.

MicroBlaze Processor Reference Guide Embedded Development Kit EDK 10.1i UG081. XILINX, 2008.

Zynq-7000 All Programmable SoC Data Sheet:Overview DS190. XILINX, 2017.

R. Yuste and T. Bonhoeffer. Morphological changes in dendritic spines associated with long-term synaptic plasticity. *Annual Review of Neuroscience*, 24(1):1071–1089, mar

2001. doi: 10.1146/annurev.neuro.24.1.1071. URL *https://doi.org/10.1146/annurev.neuro.24.1.1071*.